中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-18-003

Safe and shorter path planning for autonomous mobile robots by multi-objective island-based parallel genetic algorithm with dominating pool

Yau-Zen Chang, Kao-Ting Hung, and Jing-Sin Liu

Mar. 23, 2018(update) || Technical Report No. TR-IIS-18-003
http://www.iis.sinica.edu.tw/page/library/TechReport/tr2018/tr18.html

Safe and shorter path planning for autonomous mobile robots by multi-objective island-based parallel genetic algorithm with dominating pool

Yau-Zen Chang[b], Kao-Ting Hung[a, b], and Jing-Sin Liu[a]

[a]Institute of Information Science, Academia Sinica, Nangkang, Taipei, Taiwan 115, ROC

[b]Department of Mechanical Engineering, Chang Gung University, Tao-Yuan, Taiwan 333, ROC

zen@mail.cgu.edu.tw, kattin@st.cgu.edu.tw, liu@iis.sinica.edu.tw

Abstract*—This paper studies the application of evolutionary multi-objective optimization to path planning for mobile robots to move smoothly and safely along a shorter curvature-constrained path in completely known, planar static environments. The cost of travel is bi-objective: a new intrinsic cost of obstacle avoidance, which is designed as a weighted penetration depth to vertices of polygonal obstacles, and a length cost. The path is composed of a pre-specified number of control points, which are points of smooth turning, connected by three sub-paths composed of cubic spiral segments, where the intermediate configurations (locations and orientations) of these control points are design variables, subject to path smoothness constraint. We develop a Pareto-based evolutionary multi-objective optimization using island-based parallel genetic algorithm (IPGA) with nonsmoothness handling, aiming for searching smooth and shorter collision-free paths. To highlight the relative merit of IPGA in robustness to variations of environments, a comparative study on path planning performance based on simulations is conducted with two popular evolutionary multi-objective optimizers NSGA-II, SPEA-2 in terms of success rate in multiple runs and the shortest path length whenever a collision-free path can be successfully found. Results are presented for planar simulated rectangular environments composed of three distinct types of obstacles: polygons, walls as well as the combination of both. Our comparative study based on simulations shows that IPGA is more robust in all testing environments, while NSGA-II and SPEA-2 has a better distributed approximation to Pareto-front but sometimes the performance degrades greatly to be able to find a feasible path, especially in the environments containing wall-like obstacles.

I. INTRODUCTION

Global or local path planning of autonomous mobile robots is concerned with efficiently planning a safe path between two locations in an obstacle-rich environment.

Path planning problem can be formulated as a problem of optimization of some criteria, such as energy consumption, travel distance, travel time, subject to geometric (e.g. obstacle avoidance) and physical/motion constraints (e.g. velocity and acceleration limits). It has been studied by numerical methods, such as potential field method [1], which heavily rely on the computation of the distance from the robot to the surrounding obstacles. These methods sometimes cannot be implemented efficiently in a complex environment consisting of many obstacles of varying geometry, due to (i) obstacles may require mathematically tractable modeling [2], for example [3] used Bump surface to approximate arbitrarily-shaped obstacles; (ii) these methods lack flexibility to rapidly respond to changes in environments. This stimulates the interests in developing evolutionary search or metaheuristic methods, in particular genetic algorithm (GA) [4, 5], [38] that are able to adaptively search a large design space, for the 2D path planning problems in robotics [6-8]. Although nondeterministic nature and slowness of evolutionary path planning approaches are drawbacks to real-time operation due to reliability and efficiency concerning, it still can be implemented in virtually any environments. Moreover, GA allows multiple paths to be generated in a single run [9], which are necessary in the event that a specific path cannot be traversed so that alternative paths must be replanned. These GA-generated paths provide a range of useful paths that deserve further development for a variety of practical application needs.

For ease of implementation, the turning constraint, i.e. smoothness or curvature-continuity of the tracked path by a mobile robot is essentially important [10], usually unaccounted for in many researches [38]. Nonsmooth motions may cause slippage of wheels which degrades the robot's dead reckoning ability. When no

orientation is considered, there have been some works applying the genetic algorithms to plan a path for a point robot to traverse along a series of collision-free connected line segments linking initial position and target position [11-17]. A path in this case is encoded as a set of intermediate nodes between the start and the target in a static, known 2D grid environment, where both free space and obstacles are discretizes as a collection of grids of equal size. Some of these evolutionary planners contain many specially designed ("path repair") operators tailored for the path planning problem (e.g. [13]), which are essential for obtaining good paths under evolution. These path planning problem-specific operators aimed for getting a solution seem too many in number, and are not generally applicable.

In reality, the path planner of a mobile robot must not only be able to generate a collision-free path but also a path that meets other criteria, such as path smoothness, minimum length or minimum energy. This yields an evolutionary multi-objective optimization [18-22] formulation to path planning problem with the advantage that multiple tradeoff solutions can be obtained in a single run. Fujimura [23] used Pareto-optimality [18] to plan the paths with minimizing path length and energy consumption. [24] integrated a fuzzy tournament selection into a multi-objective evolutionary path planner that minimizes the piecewise linear path length, some measure of variations in path segment slopes. [25] integrated a probabilistic graph construction algorithm with an evolutionary multi-objective optimizer for piecewise linear path planning of a mobile robot. [26] implemented the Pareto-based GA with elitist replacement strategy to resolve the offline point-point path planning problem in a grid environment in a more satisfactory manner. According to our recent works on smooth

cubic spiral path planning via evolutionary search [27-29], we explore that the advantage of using parallel genetic algorithm based on the island model (abbreviated as IPGA) in the generation of multiple smooth collision-fee paths of mobile robots, which are modeled as a unicycle. Specifically, a three island PGA is used and the path is composed of a pre-specified number of cubic spiral segments, which inherently encodes the curvature-continuity constraint into the candidate paths.

Robustness to a variety of environments and variation in start and goal configurations is a desirable feature of mobile robot path planning algorithms. In this paper, an evolutionary path planner for mobile robots based on IPGA with dominating pool (IPGADP), aiming for improvement of the performance of IPGA revealed in our past work, is developed to plan a multi-segment cubic spiral path through a pre-specified, ordered set of control points. These control points are the control variables to be searched with a tradeoff in the traversal cost, where avoidance of static obstacles and path length are in a dilemma, i.e. a two-objective optimization problem. Additionally, a new intrinsic cost incorporating penetration depth of obstacles is designed to assist discrimination of which paths are closer to collision-free. The non-dominated sorting genetic algorithm (NSGA-II) [20, 21] and strength-Pareto evolutionary algorithm (SPEA-2) [22], both are popular Pareto-based approaches in multi-objective evolutionary algorithms (MOEAs), are implemented for a comparative study of performance gain in terms of success rate (the number of successful runs in multiple runs that find at least a smooth collision-free path) and shortest path ever successfully found. Three distinct types of obstacles: polygons, walls and their combinations are tested.

Furthermore, path planning in the dynamic environments is indeed a challenging

problem. Many other researches solved the dynamical problems with the numerical methods [30] and evolutionary methods [31, 32], [39]. These researches solve the path planning problem with complicated mathematical computations and the planned paths are non-smooth since they are composed of piece-wise line segments. In this paper, we present an evolutionary path planning approach to plan a collection of smooth paths for a mobile robot with cubic spiral segments in not only static but also dynamic environments.

The remainder of the paper is organized as follows. Section II briefly reviews the cubic spiral method and introduce a new penetration-depth based intrinsic cost for obstacle avoidance. The path planning algorithms for static environments based on evolutionary multi-objective optimization with non-smoothness handling, including the proposed IPGADP, IPGA, NSGA-II, and SPEA-2, are presented in Section III. Section III also reveals the procedures that describe how these planners plan the paths within dynamic environments. Comparisons and simulations are presented in Section IV. Finally we make a conclusion in Section V.

II. PRELIMINARIES

Let $q \equiv (x, y, \theta)$ represents a configuration where (x,y) and θ denotes the position and orientation, respectively, of the mobile robot. The path followed by a unit-speed mobile robot starting from the initial configuration (x_0, y_0, θ_0) is governed by integrating the nonholonomic kinematic constraints,

$$\begin{cases} \theta(s) = \theta_0 + \int_0^s \kappa(t)dt \\ x(s) = x_0 + \int_0^s \cos(\theta(t))dt \\ y(s) = y_0 + \int_0^s \sin(\theta(t))dt \end{cases} \quad (1)$$

where x, y and θ represent the function of position in x-axis and y-axis and orientation of robot through a path, s is the path length, and is set as 0 at the initial point of robot (x_0 , y_0), as Fig. 1 depicted. The $\kappa(t)$ can be defined in the following paragraph (eqn. 3).

A. Two objectives for path optimization

In this work, the most important objective is the obstacle avoidance, whose cost can be represented by the intrinsic cost through assuming that the robot is a point to avoid time-consuming collision detection between rigid objects [33]. The so-called intrinsic cost means that how many intersections a path intersect with all obstacles, or how many sampling nodes of path locate outside the boundary of the environment. Thus, paths with zero intrinsic cost are collision-free and our previous work [27] implements this to measure the index of collision for a path.

Actually, there have some defects in the previous intrinsic cost function, i.e. only calculate how many intersections. For example, as top of Fig. 2 shown, the similar paths (solid and dash lines) go through a thin or large obstacle will always result in the same cost when we implement the original intrinsic cost definition. Obviously, the solid path is better than the dash path since the former is much easier to be collision-free. Due to the above reason, we design a new intrinsic cost function to avoid the unusual situation. As bottom of Fig. 2 shown (enlarged diagram of circle in top), there are n sampling nodes, q_1-q_n, of the path that intersected with an obstacle. For the original intrinsic cost function,

we just consider the number of nodes intersected with obstacles or located outside the map bounds, in this example, the cost should be n. For the newly designed cost function, we not only consider the number of intersections but also incorporate the concept of degree to be collision-free. In [39], the probability of intersection was proposed to tackle with the changing environment and unvertainites.

In order to preserve the concept of original cost evaluation, we define the intersected ratio for every path segment. In this work, we sample every cubic spiral segment with the same number of nodes, i.e. N_s. The intersected ratio for i-th path segment can be defined as:

$$r_i = n_i / N_S$$

where n_i represents the number of nodes that intersected with obstacle within i-th segment. In this example, if we assume that q_1-q_n locate within the same segment, the intersected ratio would be n/N_S.

Following the previous assumption, we measure the distances from the middle node q_m to all vertexes of the obstacle and select the minimum, $d_{\min}^{m}$, to be the degree of being collision-free. Please note that if the number of intersected nodes is odd, q_m would be unique. Contrarily, if the number is even, we select two median nodes, i.e. q_m and q_{m+1}, to derive two minimum distances, $d_{\min}^{m}$ and $d_{\min}^{m+1}$. It implies that the degree would be average of these two distances. In this example, the new cost would be:

$$n / N_S \cdot d_{\min}^{m}.$$

Obviously, the evaluation of new cost can easily tell which path in Fig. 2 (a) is better. The general form of new intrinsic cost function can be defined as:

$$f_{new} = \sum_{i=1}^{N_{seg}} r_i \cdot D_i$$

where N_{seg} represents the number of cubic spiral segment within a path. And the r_i and D_i can be defined as:

$$r_i = n_i / N_S$$

$$D_i = \begin{cases} \left(d_{\min}^{m}\right)_i & \text{, if } n_i \text{ is odd} \\ \left(\left(d_{\min}^{m}\right)_i + \left(d_{\min}^{m+1}\right)_i\right)/2 & \text{, if } n_i \text{ is even} \end{cases}$$

where $\left(d_{\min}^{m}\right)_i$ represents minimum distance of middle node of i-th segment. In this research, we will present the comparisons of two different cost definitions employed in the path planning problem.

The other objective is the length of path should be minimized, and this can be computed very easily via the following equation (5).

B. Review of Cubic Spiral Method

For smooth path generation, the path is made up of several cubic spiral segments, which are curvature continuous.

1). Cubic Spiral: By definition, cubic spiral is a set of trajectories that the direction function θ is a cubic polynomial of curve length l. Its angle, which describes how much the curve turns from the initial orientation to final orientation, is denoted by

$$\alpha = \theta(l) - \theta(0) \tag{2}$$

From the first equation of (1) and the boundary conditions at s=0 and s=l, we have (Lemma 2, [31]),

$$\kappa(s) = \frac{6\alpha}{l^3} s(l - s) \tag{3}$$

If the length of a cubic spiral is 1, its size is given by (Lemma3, [34])

$$D(\alpha) \equiv 2\int_0^{1/2} \cos\left(\alpha\left(\frac{3}{2} - 2t^2\right)t\right)dt \tag{4}$$

Due to similarity of all cubic spirals, the value $D(\alpha)$ can be computed and then derive the curve's length l by the following equation (Proposition 8, [34]),

$$l = \frac{d}{D(\alpha)} \tag{5}$$

where d is the distance of two configurations.

2). *Concept of Symmetric Configurations*: For an arbitrary configuration q, $[q]$ denotes its position (x, y), and (q) its direction θ. For a configuration pair (q_1, q_2), the size d is the distance between the two points $[q_1]$ and $[q_2]$, and the angle α is the deflection angle between the two orientations (q_1) and (q_2).

In [34], a symmetric mean q of any configuration pair (q_1, q_2) is a configuration that leads (q_1, q) and (q, q_2) are both symmetric pairs. All symmetric means of a configuration pair (q_1, q_2) forms a circle if $(q_1) \neq (q_2)$ or a line connecting q_1 and q_2 if $(q_1) = (q_2)$ (Proposition 3, [34]). It is noted that the symmetric property is very important in this method because a cubic spiral can connect two symmetric configurations.

3). *Original cubic spiral path planning method*

The cubic spiral method can connect two given configuration q_1 and q_2 according to the following steps:

I. If q_1 and q_2 are symmetric, connect these two configurations with a cubic spiral directly.

II. Else, connect these two configurations with a specified symmetric mean

i. If (q_1) ≠ (q_2), the symmetric mean locates on a circle formed by two given configurations, i.e. non-parallel case.

ii. If (q_1) = (q_2), the symmetric mean locates on a straight line connected by two given configurations, i.e. parallel case.

The symmetric mean q_{Sym} of two given configurations q_1 and q_2 can be defined as the following equation according to the position ratio γ:

Non-parallel case:

If (q_1) ≠ (q_2), we should define the center of the circle that go through the given configurations q_1 and q_2.

$$p_c = (x_c, y_c) = \left(\frac{x_1 + x_2 + c(y_1 - y_2)}{2}, \frac{y_1 + y_2 + c(x_2 - x_1)}{2} \right)$$

$c = \cot\left(\frac{\theta_2 - \theta_1}{2} \right)$. Thus the position of symmetric mean [q_s] can be defined as:

$$[q_s] = (x_c + r \cdot \cos(\beta_1 + (\beta_2 - \beta_1) \cdot \gamma), y_c + r \cdot \sin(\beta_1 + (\beta_2 - \beta_1) \cdot \gamma))$$

where β_1 and β_2 are represent the orientations from p_c to q_1 and q_2 respectively. The orientation of the symmetric mean can be defined according to the position of symmetric mean [34].

Parallel case:

If (q_1) = (q_2) = θ,

$$q_s = (\ x_1 + (x_2 - x)_1 \cdot \gamma\ ,\ y_1 + (y_2 - y)_1 \cdot \gamma\ ,\ \beta - (\theta - \beta)\ )$$

where [q_1] = (x_1,y_1), [q_2] = (x_2,y_2), $\beta = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1} \right)$

III. EVOLUTIONARY CUBIC SPIRAL PATH PLANNING

In this section, the representation of a candidate path for evolutionary algorithms (EAs) is described firstly. The following paragraph contains the description of different schemes of evolutionary algorithms (EA). In general, the simple genetic algorithm (SGA) has a serious problem of premature convergence, i.e. genetic drift. To promoting the diversity within the population, a variation of scheme, parallel genetic algorithms based on island model (IPGA), is developed based on the SGA in our previous work [27-29]. Actually, the IPGA only has an advantage on high success rate when dealing with some test cases in the past. To improve the IPGA, we propose the IPGA with dominating pool, abbreviated as IPGADP, with the modification of the role of migrating pool of IPGA during evolution. Additionally, we also implement the NSGA-II and SPEA-2, both are the most popular Pareto-based approaches in MOEAs, to this problem for comparison of algorithm performance.

A. Individual representation: candidate path

A path segment is defined by a continuous mapping $\tau : [0,1] \to C$ where $q(s) = (x(s), y(s), \theta(s))$ denotes robot configuration with s arc length. A path is composed of a set of path segments connected via a pre-specified number of intermediate configurations. In this paper, we use cubic spiral as a path segment for a mobile robot, which is kinematically feasible. For a given start configuration, a cubic spiral can be defined by the size d and deflected angle α, which has: its length via (5), its curvature function by (3) and its terminal configuration from equation (1). In addition, when the size of cubic spiral is negative, we can plan the backward motion of the robot according to equation (3) and (1), i.e. l should be negative.

In this paper, a path is composed by three set of subpaths that are evolvable, each is composed by several cubic spiral segments: subpaths ***S***, ***G***, ***M*** (depicted in Fig.3). The subpath ***S*** is composed of cubic spiral segments, planned forwardly from START through a prespecified number of intermediate configurations. Similarly, the subpath ***G*** is planned backwardly from GOAL. Finally, ***S*** and ***G*** are connected by two cubic spiral segments defined via a symmetric mean [34], i.e. subpath ***M***. This composing strategy can make sure that the path is connecting the given start and goal configurations. If we define the subpaths ***S*** and ***G*** by N cubic spiral segments, i.e. N control points, the chromosome would consist of $2N+1$ genes (the size and deflected angle for each cubic spiral segment in ***S*** and ***G***, and the position ratio of symmetric mean for the subpath ***M***). The composition of genes for a single chromosome θ is the ordered list, excluding the given START and GOAL configurations

$$\theta = [d_1\ \alpha_1;\ d_2\ \alpha_2;\ \ldots;\ d_N\ \alpha_N\ ;\ \gamma_{Sym}],\ N\text{: even number} \tag{6}$$

For the ease of programming, N is set as an even number so that the two subpaths ***S***, ***G*** have equal number of segments (N/2).

Due to the natural characteristic, the cubic spiral would be a spiral curve when the deflected angle is larger than a specific value. The above situation would result in a non-smooth path when we connect the subpaths ***S*** and ***G*** by ***M*** since we can't assure that the deflected angle between last nodes at subpaths ***S*** and ***G*** is smaller than the specific value, as shown by dot circle of Fig. 4 in which the shown path consists of 6 control points. It implies that the candidate path would be a smooth or non-smooth path. It also leads this work to a constraint-handling optimization problem.

B. Specifically Subgoals Manipulated Operator

A specifically manipulated operator for infeasible path is designed to increase the efficiency of searching the collision-free paths. For the mobile robot path planning problem, it is important to consider how to elaborate an infeasible path into a more acceptable path. To this aim, we further design a local path refinement operator, subgoals manipulation operator, operating on the infeasible paths segments that cross the obstacles in order to accelerate the evolution to find out the collision-free paths. The operator locally manipulates those nodes in a predefined neighborhood, so that this local refinement of path shape occurs in the sub-regions. Here the manipulation is primarily based on the mutation operator. Fig.4 shows the diagram of this operator. The second path segment of original path (solid path) crosses an obstacle, and the operator randomly shifts the configuration of Node 2 to a new neighboring location to obtain new path (dashed path), very possibly becoming collision-free.

In this work, all schemes of MOEAs are incorporated with this operator. The number of manipulations for each run is defined as 10% of populations/subpopulations, and the infeasible paths with lower intrinsic cost will be selected firstly.

C. The Island-based Parallel Genetic Algorithms (IPGA) [27-29]

In the evolutionary algorithms, the conflict between speedup of convergence rate and avoidance of local minimum is embodied in the selective pressure and population diversity. With a single population, as in the SGA, the selective pressure and population diversity oppose with each other. By IPGAs, it is possible to raise selective pressure in some subpopulations and concurrently to augment the population diversity in other

subpopulations. Performance of the IPGA is affected by four factors: number of migrants, migration interval and the selection and replacement strategy of individuals. In this work, the migration between different subpopulations is activated every predefined generation. The following paragraphs briefly describe the detail of the IPGA.

1). Fitness Definition: Rank-based Assignment & Pareto Ranking

For the IPGA, we apply the fast non-dominance sorting method proposed by [21] to rank the individuals in a population. Higher ranks will be given higher indices. The solutions of highest rank are termed as Pareto-frontier.

2). Selection

The roulette-wheel selection operator is employed. For faster convergence, elitism is used to retain some preferred individuals at each generation. In general, evolutionary searching can be accelerated with the preservation of elitisms, and the preserved elitisms have great pressure to influence the evolutionary results. One of the most popular criteria to select representative solutions from the Pareto-frontier is the min-max method [36]. The main idea of this method is to select a point within the two ends of Pareto-frontier that the maximum deviation of objectives is minimized. In reality, in order to increase the stability of searching, we preserve at most two solutions with the min-max method at each generation. Besides, in order to cover the whole frontier, we also prefer to preserve the solutions with extreme values along every dimension.

3). Genetic Operation: Crossover and Mutation

The crossover is implemented in IPGA as a linear interpolation between two chromosomes, also called arithmetic crossover [37]. The following equation shows the

interpolation operation between two distinct chromosomes:

$$\begin{cases} \theta_1' = \theta_1 \cdot \gamma_1 + \theta_2 \cdot (1 - \gamma_1) \\ \theta_2' = \theta_2 \cdot \gamma_1 + \theta_1 \cdot (1 - \gamma_1) \end{cases},$$

where γ_1 is a randomly generated real number between 0 and 1. Please note that γ_1 is different in different genes.

Mutation is a mechanism introduced to explore new searching directions. Assuming $\theta_{\max}$ and $\theta_{\min}$ be the bounds of candidate solutions, the resultant descendant generated by the mutation operation will be:

$$\theta' = \theta_{\min} + \gamma_2 \cdot (\theta_{\max} - \theta_{\min}),$$

where θ stands for a chromosome in a population, γ_2 is a random number between 0 and 1.

4). Migrating policy

For the migration during evolution, there exists a common pool to mix the migrated chromosomes (to the pool) from each island. Actually, these migrated individuals are selected based on their ranks and the immigrated individuals (to islands) from common pool are selected randomly.

5). Constraint handling

In this work, the constraint is very difficult to be reflected in the design space, thus we only can make a boolean discrimination for every individuals. This will results in two categories of population at every generation, i.e. constraint satisfied and non-satisfied solutions. For the proposed IPGA, we generate the child population according to selection based on two rankings: first is ranking for total populations, second is ranking only for constraint satisfied solutions. Actually, the purpose of first ranking is to make the

evolution to preserve the diversity of searching. In our implementation, these two rankings have the same probabilities to be used to select children.

D. Non-dominated sorting genetic algorithm (NSGA-II)

The original non-dominated sorting genetic algorithm (NSGA) was proposed by Deb in 1995 [20]. Unfortunately, there had some criticisms like high computation complexity, lack of elitism and need to specify a parameter for sharing function. At 2002 [21], the authors alleviated all the above difficulties and proposed a new approach termed as NSGA-II. NSGA-II adopt a fast non-dominated sorting approach to reduce the computed complexity resulted by the sorting. On the other hand, NSGA-II replaces the sharing function with a crowded comparison operator to improve searching efficiency. In order to preserve the elitisms, NSGA-II needs to generate a new child population through selection, crossover and mutation, finally combine with the parents. Since the size of combined population would be twice of population size, it is needed to be extracted to an elitist population according to the sorting and crowded comparison operator. This mechanism also assures that the elitisms would be preserved during evolution. The crowded comparison operator guides the evolution to find out the uniformly spread-out Pareto-frontier through the computation of crowding distance. Here the crowding distance tries to estimate the density of solutions surrounding a specific solution. The crowding distance is defined as the average distance of two neighbors of the specific solution along every objective. They also proposed a constraint handling approach for NSGA-II, termed as Constrained NSGA-II. Actually, this approach is modified from original NSGA-II for the dominance checking between two individuals. Please note that the checking procedure considers the constraint. In detail, there are three cases when checking dominance: two individuals are constraint satisfied solutions, two solutions are both constraint non-satisfied, and one is constraint

satisfied and the other is not. The first and third cases are easy to define the dominance. The second needs to compute the critical values of constraint violation for two individuals and check the dominance according to these two values. As we mentioned in Section III.C.4), we only can discriminate that the individual is constraint satisfied or not. Actually, NSGA-II dramatically improves the capability to find out real Pareto-frontier of the given artificial problem no matter it involves with constraint or not.

E. Improved Strength Pareto Evolutionary Algorithm (SPEA-2)

The original strength Pareto evolutionary algorithm (SPEA) was proposed in 1995. An additional archive and a regular population were evolved under the proposed steps. The main idea of this algorithm is the strength Pareto ranking. The authors proposed a special fitness assignment strategy for an individual that considers not only how many individuals it dominates and it is dominated by. Actually, there are three potential weaknesses which can be addressed as fitness assignment, density estimation, and archive truncation. In 2001 [22], the authors proposed an improved version of SPEA (SPEA-2) that revised the above disadvantages. SPEA-2 assigns the fitness for individuals with minor modification of original idea and the density information. Furthermore, the size of archive is fixed and the clustering technique retains the characteristic of the Pareto front without losing the boundaries of it for the archive. On the other hand, the recombination and mutation operators are applied to the mating pool that is formed via binary tournament selection on the archive and finally forms the new regular population.

F. IPGA with dominating pool (IPGADP)

According to our past works, the proposed IPGA indeed had high success rates on

planning a feasible path than the other MOEAs. Nevertheless, the found paths sometimes have poor quality in the length of criterion and it might be resulted by the preservation of only four solutions within the Pareto-front. On the other hand, NSGA-II and SPEA-2 have the better performance in this aspect but they do have heavier computations than the IPGA due to entire preservation of population, crowding distance computation in NSGA-II, and clustering computation in SPEA-2. For avoiding the heavy computation, IPGA with dominating pool (IPGADP) is proposed to optimize the path planning problem more efficiently. The following paragraphs will introduce the details of IPGADP.

1). Basic scheme of IPGADP

The IPGADP is a modification of the IPGA with the alternation of the function of migrating pool. In IPGA, the migrating pool is used to make each identical island to have the chance to communicate with each other and the evolutions are proceeded at each island independently. In IPGADP, the pool plays a dominating role in evolving the subpopulations at each island. Specifically, the new subpopulations are mainly bred from the dominating pool and their original subpopulations instead of evolving at its own island and communicating with the migrating pool. In this work, the chromosome is selected to be recombined and mutated with higher probability from the pool than from the original subpopulations (0.8 and 0.2 respectively).

2). Dominating pool

The purpose of the IPGADP is to determine the global Pareto-optimal front as behaviors of NSGA-II [21] and SPEA-2 [22] but without a heavy computation. For the

IPGADP, a dominating pool is assigned to gradually locate the global Pareto-optimal solutions of the given problem during the optimization. Specifically, the dominating pool will alter its size and locate the global Pareto-optimal front when the subpopulations breed the local Pareto-optimal solutions. As mentioned in [21], a book-keeping strategy is implemented to efficiently define a non-dominated set from a group of solutions. The dominating pool is depicted as P_{dp} and the local Pareto-optimal solution at each island is represented as s. For each local Pareto-optimal chromosome, the procedure of altering on the pool P_{dp} can be shown as follows,

```
For each chromosome s_i within the subpopulation
   If P_dp is null, includes s_i into P_dp
   Else,
        For each chromosome s_i' within the pool P_dp
             If s_i.Cost dominates s_i'.Cost, then removes s_i' from P_dp
        End For
        If s_i.Cost is not dominated by any s_i'.Cost, then includes s_i into P_dp
End For
```

where .*Cost* represents the objective function values of the chromosome. As the former description, the dominating pool will gradually collect the found trade-off solutions during the optimization. Furthermore, the dominating pool should be limited within a predefined number of sampling points since the growing procedure will be time-consuming if the number of elements within the front is large. Based on the crowding distance computation [21], the size of the pool is limited within a predefined number of chromosomes. Basically, the assignment of the crowding distance is calculated according to the normalized distances of the neighbors next to a specific point within objective space. For a chromosome s_i, the crowding distance can be defined as the sum of the normalized distances between the neighbors along all dimensions, just like the following description,

For a sorted group s_i ($i = 1, 2, \ldots, n$) of the dominating pool p_g

If $i = 1$ and n, then $s_i.CrowdDist = \infty$

If $i = 2, 3, \ldots, n\text{-}1$, then $s_i.CrowdDist = \sum_{j=1}^{m} \frac{\left| s_{i+1}.Cost_j - s_{i-1}.Cost_j \right|}{\left| s_n.Cost_j - s_1.Cost_j \right|}$

End For

where $s_i.CrowdDist$ represents the crowding distance of a chromosome s_i, m stands for the number of the objective, $s_i.Cost_j$ means the j-th objective function value of s_i. The size of p_{dp} can be limited by sorting the group according to their crowding distance and eliminating those with small distances. Obviously, the end points of growing front will be selected since they have the extremely large crowding distances with respect to other points. Please note that the size of dominating pool is half number of total populations in this work.

3). Genetic operators

In IPGADP, the binary tournament selection is used to select the parent chromosomes from those identical subpopulations or dominating pool to breed new children. For the subpopulation, the fast non-dominated sorting [21] assigns ranks for every chromosomes and the selection is proceeded based on this ranking (better solutions have more chance to be selected). Additionally, the chromosomes within the dominating pool will be assigned the corresponding crowding distances [21] and this can be used to be the basis of selection operator (sparse solutions have more chance to be selected). Finally, the crossover and mutation operator is the same with NSGA-II.

4). Constraint handling

Different with IPGA, the constraint handling of the IPGADP is the same with NSGA-II, i.e. constraints are considered in dominance checking as describing in Section III.D.

5). Pseudo code of IPGADP

The following pseudo code describes the procedure of the IPGADP.

```
τ = 0.                  // τ: the generation count
Initialize P_i(τ).      // P_i(τ): subpopulation at generation τ at i-th island
Initialize P_dp.        // P_dp: dominating pool (it is null at initialization)
While (termination condition is not satisfied) do
    For i = 1 to the number of islands
        Non-dominated sorting on P_i(τ) and derives f_nds(P_i(τ)).
        Adopt the book-keeping strategy (Section III.F.2) to check the
          local front in P_i(τ) to alter the dominating pool P_dp.
        Crowding distance computation (Section III.F.2) on P_dp and
          derives f_cdc(P_dp).
        While size of P_i(τ+1) < size of P_i(τ) do
            If random number < 0.8, select father θ_i1 from P_dp based on
              f_cdc(P_dp); otherwise, select father θ_i1 from P_i(τ) based on
              f_nds(P_i(τ)).
            If random number < 0.8, select mother θ_i2 from P_dp based on
              f_cdc(P_dp); otherwise, select mother θ_i2 from P_i(τ) based on
              f_nds(P_i(τ)).
            Perform genetic operations on parents θ_i1 and θ_i2 to generate
              children θ_i1' and θ_i2' according to crossover rate ϖ_c(i) and
              mutation rate ϖ_m(i).
            Put θ_i1' and θ_i2' into P_i(τ+1).
        End While
    End For
    τ = τ + 1.
End While
Report the dominating pool P_dp as the result.
```

For the path planning problem in this work, the subgoal manipulated operator is embedded before the breeding of next subpopulation. The crossover and mutation rates ϖ_c and ϖ_m of each identical island can be defined

G. Intrinsic cost for known moving obstacles

As there exist moving (translating, rotating or their combination) obstacles in the environment, it is assumed that the motion trajectory of each obstacle is completely known. The determination of intrinsic cost of a path in dynamic environment is detailed as follows:

Step1: A given candidate path is sampled to obtain a collection of sampling points $\{q_i\}$. By (1), the passing time sequence $\{t_i\}$ is derived for a mobile robot moving with unit-speed when it traverses the sampling points $\{q_i\}$ along the path. Note that different path length will yield different time sequence and thus different traveling time.

Step2: For the time sequence $\{t_i\}$, compute the locations of all moving obstacles with completely known motion.

Step3: One by one check all sampled path points $\{q_i\}$ or equivalently the time instants $\{t_i\}$ whether the robot crossed any dynamic obstacles at some time instants. Sum the intrinsic costs for intersecting moving obstacles at those time instants of collision $\{t_{ci}\} \subset \{t_i\}$. This sum is the intrinsic cost of the given path resulting from moving obstacles.

Step4: For static obstacles, the intrinsic cost of the given path is computed as usual.

Step5: The total sum of intrinsic costs computed in Step3,

Step4 is the intrinsic cost of the candidate path mentioned before. Note that at a given instant t_i, the robot moving along a path can encounter at most one obstacle. An essential difference between intrinsic cost calculation of static and dynamic obstacles is as follows. If the robot encountered a specific moving obstacle denoted as o at multiple time instants

$$\left\{t_{ci}^{o}\right\} \subset \left\{t_{ci}\right\} = \oplus_{\text{for each obstacle } o} \left\{t_{ci}^{o}\right\},$$

where $\oplus$ denotes the direct sum, the intrinsic cost for that obstacle will be the sum of intrinsic costs calculated at all time instants $\left\{t_{ci}^{o}\right\}$ of collision that robot crossed the

obstacle o.

IV. Experimental Results

In this section, simulations in either static or dynamic environments are performed for empirical comparison of various schemes to make clear which scheme performs better. NSGA-II and SPEA-2 indeed perform better result in searching the real Pareto front of the given artificial problems. In this research, these two famous evolutionary techniques will be implemented to the path planning problem and compared to the proposed PGA. In our simulations, the implementation of path planner equipped with NSGA-II totally follows the description of [21], including the setting of parameters of crossover rate, mutation rate, and distribution indices for crossover and mutation operators. Since the original SPEA-2 didn't tackle the constraint handling, we implement the dominance checking method of NSGA-II to consider the constraint of the path planning problem. Furthermore, the crossover and mutation operators are the same with NSGA-II for ease of comparison.

Section IV.A and IV.B are tested for static environments (Fig.5 and Fig.8) and Section IV.C is tested for dynamic cases (Fig.9). We assume that the environment is rectangular and all information of obstacles, including its shapes, vertices, moving velocities, rotating speeds, and etc., is completely known. The initialization of elements of candidate path in evolution respects the following predefined range:

size d: [diagonal distance of map/ (N*2), diagonal distance of map/ (N/2)];

deflected orientation α: [$-\pi$, π];

position ratio of symmetric mean γ: [0.2, 0.8]

A. Comparison of different intrinsic cost functions with simple genetic algorithm (SGA)

In this research, we propose a new intrinsic cost function to avoid the unexpected circumstances like Section II.A mentioned. Two intrinsic cost functions are employed by SGA. The parameters definition of SGA and comparison results can be tabulated as Table 1. Each environment is proceeded with 20 independent runs and the initial population are all the same. For every single run, the evolution would be ceased until the first collision-free path is found or the maximum generations are reached. For ease of observation, the best 10 individuals of each run are extracted. The number of intersected nodes and sum of minimum distance (depicted as Section II.A) of these 200 individuals can be defined. Fig. 6 illustrates the distribution of these two values of the extracted individuals for Environment-2. Actually, there are three performance indices being considered: The success rate represents number of runs the evolution can find out at least one collision-free path before 50 generations. Averages and standard deviations computed along the above two values' dimension. We can discover that the new intrinsic cost evaluation can make the evolution perform well in success rate. Although the average and STD along number of intersections are quite close in two cost definition for three testing maps, the new cost has great improvement when we consider the last performance index. It implies that the new cost evaluation help the evolution to have more chance to find the collision-free paths.

B. Comparison for IPGA with different migration intervals, NSGA-II, SPEA-2, and proposed IPGADP

In this section, IPGA with different migration intervals, NSGA-II, SPEA-2 and IPGADP are implemented to solve this problem. Before implementation, NSGA-II and SPEA-2 are tested with two testing functions proposed by the original papers. First one is a bi-objective function without constraints but with large design space (100 design variables) and the other is also a bi-objective function but it tackles with constraints and two design variables. These two problems are solved according to the descriptions of the papers and Fig. 7 demonstrates the final optimal fronts of two cases by two different techniques. It's observed that these two have similar performances on problems either with or without constraints. Due to the NSGA-II need to define a population size that is multiple of 4 (due to tournament selection), we increase the size to 120.

In this work, there are four indices (S, L_{best}, L_{worst}, $A\pm\sigma$) being considered for algorithm performance: the success rate (S) represents the number of specific runs that find out at least one feasible path after 20 independent runs. The best (L_{best}) and worst (L_{worst}) path means the path with minimum and maximum length within those constraint satisfied paths in every single run after 20 runs. The average length (A) and standard deviation (σ) are derived from all success runs after 20 independent runs. Following the results of Section IV.A, two intrinsic cost evaluations would be tackled separately in different schemes of MOEAs, i.e. each identical scheme with two cost evaluations was proceeded with 20 independent runs. Table 2 reveals that the new intrinsic cost function also help the multi-objective evolution make better performance indices than the old cost evaluation for all environments and all schemes of MOEAs. Especially for the SPEA-2,

the performance of SPEA-2 is the poorest when we compare with the other schemes on dealing with the old cost definition.

For the migration intervals of IPGA, the IPGA with 5 migrating intervals (denoted as IPGA-5) has better result than the others, especially in the average length and STD of best paths. Although the IPGA-1 and IPGA-5 have close indices in average length and STD in Environment-3, the success rate of IPGA-5 are dramatically better than IPGA-1. Altogether, less of migrating intervals of IPGA scheme might invoke the convergence of the population too quickly (genetic drift). It is obvious that NSGA-II and SPEA-2 perform better result in different environments (Environment-1 and 2). On the contrary, IPGA-5 exhibits more stable performance in all maps. For the IPGADP, it indeed has our attention on its performance of path planning problem especially all performance indices are dramatically better than the other MOEAs.

For advanced comparison, we designed another three obstructed environments mixed with wall-like and polygonal (Fig. 8) obstacles to test IPGA-5, NSGA-II, SPEA-2, and IPGADP with new intrinsic cost function. Table 3 summarizes this comparison. Actually, NSGA-II and SPEA-2 indeed have similar performance. For the IPGA, the success rates for each map are still more stable than the others except IPGADP. In these three maps, the IPGADP still performs better than the other MOEAs. Although IPGADP has similar success rate with the others in Environment-5, the average length and corresponding STD of found paths are shorter and more stable than the other three schemes.

C. Comparison of different MOEAs in dynamic environments

Table 4 reveals the comparison result of different scheme of MOEAs implemented to plan paths in dynamic environments based on different intrinsic cost definition. For the

different intrinsic cost functions, the first two dynamic testing maps have similar success rates except the average feasible length of paths, i.e. the planned paths are shorter when we implement the new intrinsic cost. For the third environment, new cost definition dramatically improves the success rate of planning.

Among these testing environments, the IPGADP has the better performance than the others perform especially in the third case. The difficulty of third environment might be two rotating wall-like obstacles located in the neighbours of START and GOAL. Even though the difficulty might make the other MOEAs have poor performance, IPGADP still has good success rate and average length of found paths.

D. Summary of simulations

Before summary of the proposed simulations, we proposed some observations about the previous comparisons. Fig. 10 illustrates the Pareto-fronts of four approaches (IPGA, NSGA-II, SPEA-2, and IPGADP) from one of 20 runs at some particular generations (1, 40, 80, …, and 200) within Environment-5. The Pareto-frontiers obviously evolve to the left-bottom parts of objective space. Actually, NSGA-II has good capability to search for the real Pareto-frontier resulted by the tackled problem. Fig. 10 also illustrates that the final populations of NSGA-II, SPEA-2, and IPGADP forms the Pareto-front more diversely than IPGA. Furthermore, we can discover that different fitness assignment strategies would result in different evolutions especially the SPEA-2 has entirely different strategy with IPGA, NSGA-II, and IPGADP (they implemented the fast non-dominated sorting [21]). As Fig. 11 depicts, two special cases of ranking results demonstrate that the strength Pareto ranking assigns individuals within the same rank but with large variance.

Fig. 12 illustrates a ranking example with 500 randomly generated paths in static Environment-1, and it looks very similar to the case of Fig. 11(b). Fig. 13 also shows the evolutionary histories of planned path with these approaches from one of 20 runs within Environment-6. For ease of observation, we only illustrate the paths at particular generations. Please note that the best path at a generation should be the one that is both collision-free and with minimum length if there have feasible paths. If no collision-free path is found, the individual with minimum intrinsic cost within the Pareto-front would be the best. As you can see, the path is evolved to avoid the obstacles under the evolution. Fig.14 illustrates the planning results in another two static and larger environments with more obstacles. For the more challenging dynamic environment, Fig. 15 demonstrates a collision-free path planned by the multi-objective evolutionary planner IPGADP in a dynamic environment. There are two rotating wall-like obstacle, four moving polygonal obstacles, and two static large scale wall-like obstacles in this environment.

In all simulations of completely known environments with either static or moving obstacles, the evolutionary path planner based on IPGA, NSGA-II, SPEA-2, and IPGADP successfully generates a smooth, safe and shorter path for mobile robots without complicated mathematical computation. The population size of this research is quite small, thus we might increase the size to derive a better results when we implement these approaches. On the other hand, since the SPEA-2 and NSGA-II both have higher computation complexity due to the sorting of combined/merged populations, IPGADP would solve the problem more efficiently.

V. CONCLUSION

In this paper, IPGA with a dominating pool (IPGADP) are recommended to efficiently and more successfully solve a bi-objective optimization problem formulating to exploring potential feasible paths for a mobile robot with considering two objectives, the intrinsic cost and length of a path either in static or dynamic environments. On the contrary, the NSGA-II and SPEA-2 perform better in searching the real Pareto-front for the encountered problems. These proposed approaches are effective and flexible to generate a collection of collision-free paths to smoothly and safely move a mobile robot from a start configuration to a goal configuration in an obstructed environment. Here the cubic spiral segments are used to compose a smoothly directed planar curve. Actually, the special manipulator and preservation strategy of elitisms, as justified by simulations, effectively improve the evolutionary search capability. Furthermore, the intrinsic cost function that considers the degree to be collision-free can dramatically improve the evolution that concerns with the collision of a path. In this work, we successfully implement the evolutionary search to plan a smooth path composed of a prescribed number of cubic spiral segments in completely known rectangular environments with static as well as polygonal and wall-like obstacles.

REFERENCES

[1] J.C. Latombe, *Robot motion planning*, Kluwer Academic Publisher, New York, 1991.

[2] J. Minguez, L. Montano and J. Santos-victor, "Abstracting vehicle shape and kinematics constraints from obstacle avoidance methods," *Autonomous Robots*, Vol. 20, pp.43-59, 2006.

[3] P.N. Azariadis and N.A. Aspragathos, Obstacle representation by Bump-surfaces for optimal motion planning, *Robotics and Autonomous Systems*, Vol.51, pp.129-150, 2005.

[4] D.E. Goldberg, *Genetic Algorithms in Search, optimization, and Machine Learning*, Addison-Wesley, 1989.

[5] M.F. Man, K.S. Tang, and S. Kwong, "Genetic algorithms: concepts and applications," *IEEE Transaction on Industrial Electronics*, Vol.43, No.5, pp.519-533, 1996.

[6] K. Sugihara and J. Smith, Genetic algorithms for adaptive motion planning of an autonomous mobile robot, *IEEE International Conference on Computational Intelligence in Robotics and Automation*, pp.138-146, 1997.

[7] Y. Davidor, *Genetic Algorithms and Robotics*, World Scientific, Singapore, 1991.

[8] I.K. Nikolos, K.P. Valavanis, N.C. Tsourveloudis, and A.N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *IEEE Transactions on Systems, Man, and Cybernetics- Part B: Cybernetics*, Vol.33, No.6, pp.898-912, 2003.

[9] C. Hocaoglu and A.C. Sanderson, "Planning multiple paths with evolutionary speciation," *IEEE Transactions on Evolutionary Computation*, Vol.5, No.3, pp 169-191, 2001.

[10] A. Scheuer and Th. Fraichard, "Continuous-curvature path planning for car-like vehicles," *1997 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.997-1003, 1997.

[11] P. Wide and H. Schellwat, "Implementation of a genetic algorithm for routing an autonomous robot," *Robotica*, Vol. 15, pp 207-211, 1997.

[12] M. Gemeinder, and M. Gerke, "GA-based path planning for mobile robot systems employing an active search algorithm," *Applied Soft Computing*, Vol.3, pp.149-158, 2003.

[13] J. Xiao, Z. Michalewicz, L. Zhang, and K. Trojanowski, "Adaptive Evolutionary Planner/Navigator for Mobile Robots," *IEEE Transactions on Evolutionary Computation*, Vol.1, No.1, pp 18-28, 1997.

[14] A. C. Nearchou, "Path planning of a mobile robot using genetic heuristics," *Robotica*, Vol.16, pp 575-588, 1998.

[15] K. Sugihara and J. Smith, Genetic algorithms for adaptive motion planning of an autonomous mobile robot, *IEEE International Conference on Computational Intelligence in Robotics and Automation*, pp.138-146, 1997.

[16] J. Tu and S.X. Yang, Genetic algorithm based path planning for a mobile robot, *2003 IEEE International Conference on Control and Automation*, pp.1221-1226, 2003.

[17] Y. Wang, D. Mulvaney and J. Sillitoe, Genetic-based mobile robot path planning using vertex heuristics, *2006 IEEE International Conference on Cybernetics and Intelligent Systems*, pp.463-468, 2006.

[18] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computing*, Vol.3, No.1, 1995, pp. 1-16.

[19] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*, West Sussex, England, John Wiley & Sons, 2001.

[20] N. Srinivas and K. Deb, "Multi-objective function optimization using non-dominated sorting genetic algorithm," *Evolutionary Computation*, Vol. 2, pp. 221-248, 1995.

[21] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, Vol. 6, No. 2, pp. 181-197, 2002.

[22] E. Zitzler, M. Laumanns, and L. Thile, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," *Computer Engineering and Communication Networks Lab (TLK), Swiss Federal Institute of Technology Zurich*, Technical Report 103, May, 2001.

[23] K. Fujimura, "Path planning with multiple objectives," *IEEE Robotics and Automation Magazine*, pp.33-38, March 1996.

[24] G. Dozier, S. McCullough, A. Homeifar, E. Tunstel and L. Moore, "Multiobjective evolutionary path planning via fuzzy tournament selection," *IEEE World Congress on Computational Intelligence*, pp.684-689, 1998.

[25] V.A. Spais and L.P. Petrou, Multiobjective moton planning for a nonholonomic vehicle, *2003 IEEE Congress on Evolutionary Computation*, pp.2058-2065, 2003.

[26] O. Castillo, L. Trujillo and P. Melin, "Multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots," *Soft Computing,* Vol.11, No. 3, pp.269-279, 2007.

[27] K.T. Hung, J.S. Liu, and Y.Z. Chang, and Yau-Zen Chang, "Generation of multiple cubic spiral paths for obstacle avoidance of a car-like mobile robot using evolutionary

search," *The 3rd International Conference on Autonomous Robots and Agents*, Palmerston North, New Zealand, Dec. 12-14, 2006.

[28] K.T. Hung, J.S. Liu, and Y.Z. Chang, "A comparative study of smooth path planning for a mobile robot by evolutionary multi-objective optimization," *2007 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Jacksonville, Florida, USA, Jun. 20-23, 2007.

[29] J.S. Liu, K.T. Hung, and Y.Z. Chang, "Offline smooth mobile robot path planning in dynamic environments using evolutionary multi-objective optimization," *2008 IEEE SMC International Conference on Distributed Human-Machine Systems*, Athens, Greece, Mar. 9-12, 2008.

[30] A. Farinelli and L. Iocchi. Planning Trajectories in Dynamic Environments Using a Gradient Method, *RoboCup 2003*, LNAI 3020, pp. 320-331, 2004.

[31] R. Smierzchalski and Z. Michalewicz. Path Planning in Dynamic Environments, *Computational Intelligence*, Vol. 8, pp 135-153, 2005.

[32] H. Q. Min, J. H. Zhu, and X. J. Zheng. Obstacle Avoidance with Multi-Objective Optimization by PSO in Dynamic Environment, *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, Aug. 18-21, 2005.

[33] K. Konolige, "A gradient method for real-time robot control," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.639-646, 2000.

[34] Y.J. Kanayama, B.I. Hartman, "Smooth local path planning for autonomous vehicles," *International Journal of Robotic Research*, Vol.16, No.3, pp 263-283, 1997.

[35] T.C. Liang, J.S. Liu, G.T. Hung, and Y.Z. Chang, "Practical and flexible path planning for car-like mobile robot using maximal-curvature cubic spiral," *Robotics and Autonomous System*, Vol. 52, pp. 312-335, 2005.

[36] A.D. Belegundu and T.R. Chandrupatla, *Optimization Concepts and Applications in Engineering*, New Jersey, Prentice Hall, 1999.

[37] H.K. Lam, F.H. Leung, P.K.S. Tam, "Design and stability analysis of fuzzy model-based nonlinear controller for nonlinear systems using genetic algorithm" *IEEE Transactions on System, Man and Cybernetics, Part B*, Vol. 33, pp. 250-257, 2003.

[38] V. Pshikhopov. *Path Planning for Vehicles Operating in Uncertain 2D Environments*. Butterworth-Heinemann, 2017.

[39] A. Pongpunwattana & R. Rysdyk. Evolution-based dynamic path planning for autonomous vehicles. In *Innovations in Intelligent Machines-1* (pp. 113-145). Springer, Berlin, Heidelberg, 2007.

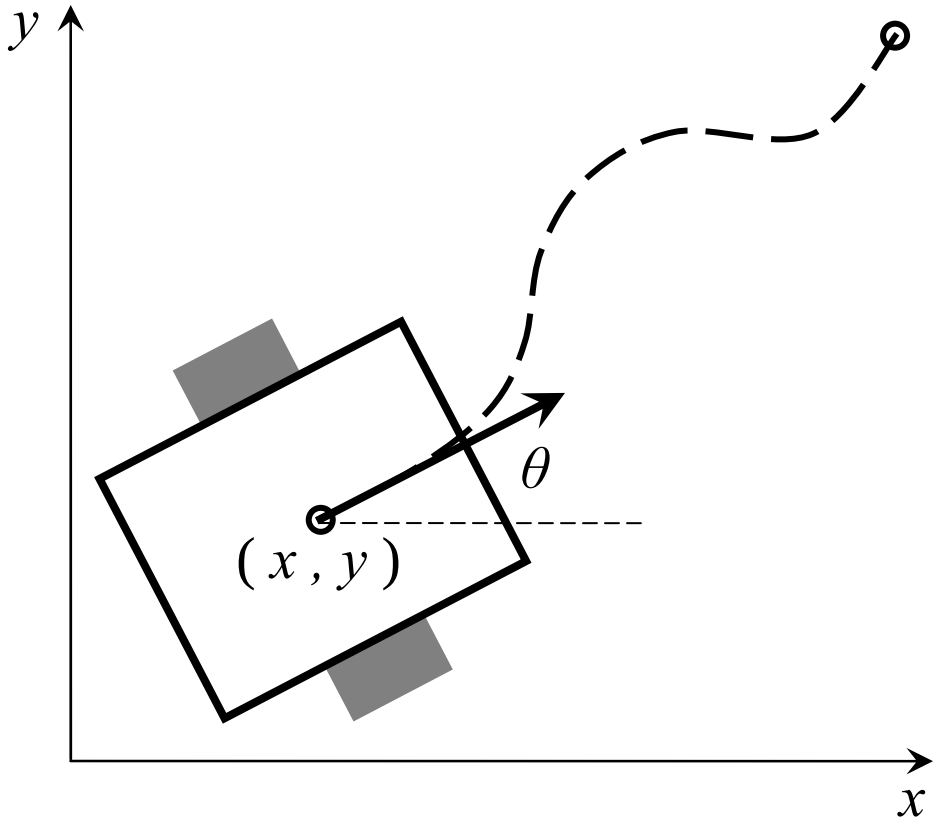


Fig. 1 Schematic illustration of path representation of the mobile robot

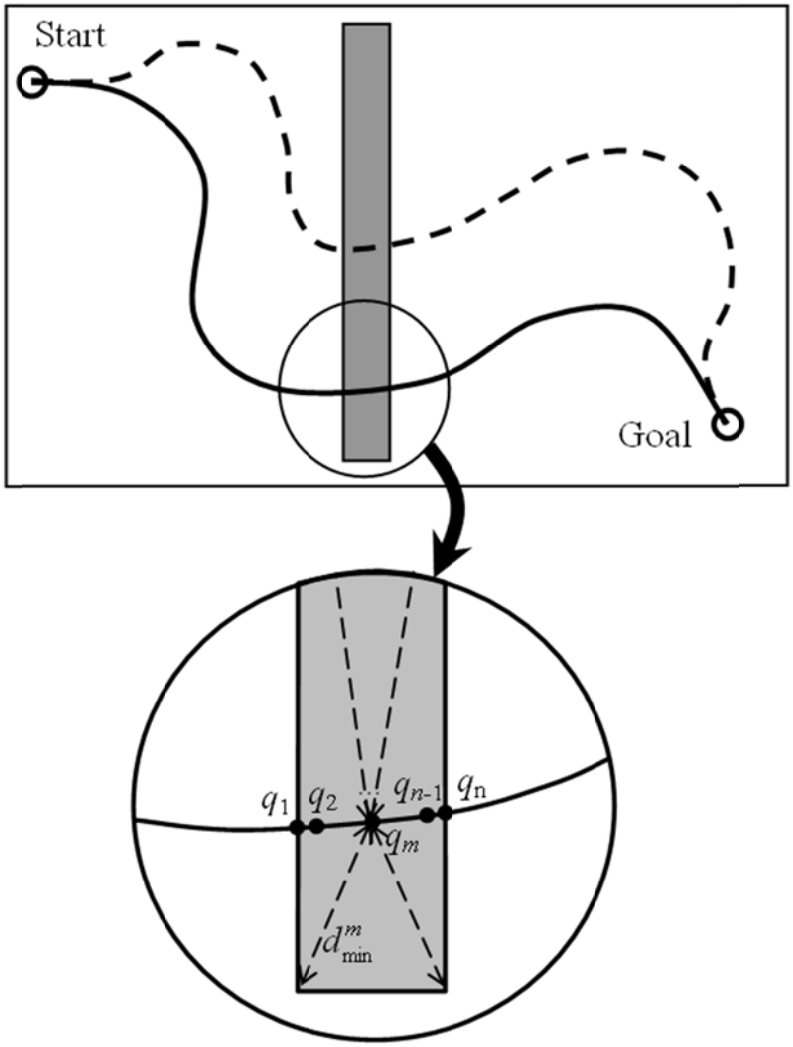


Fig. 2 Illustration of similar paths intersected with a thin obstacle

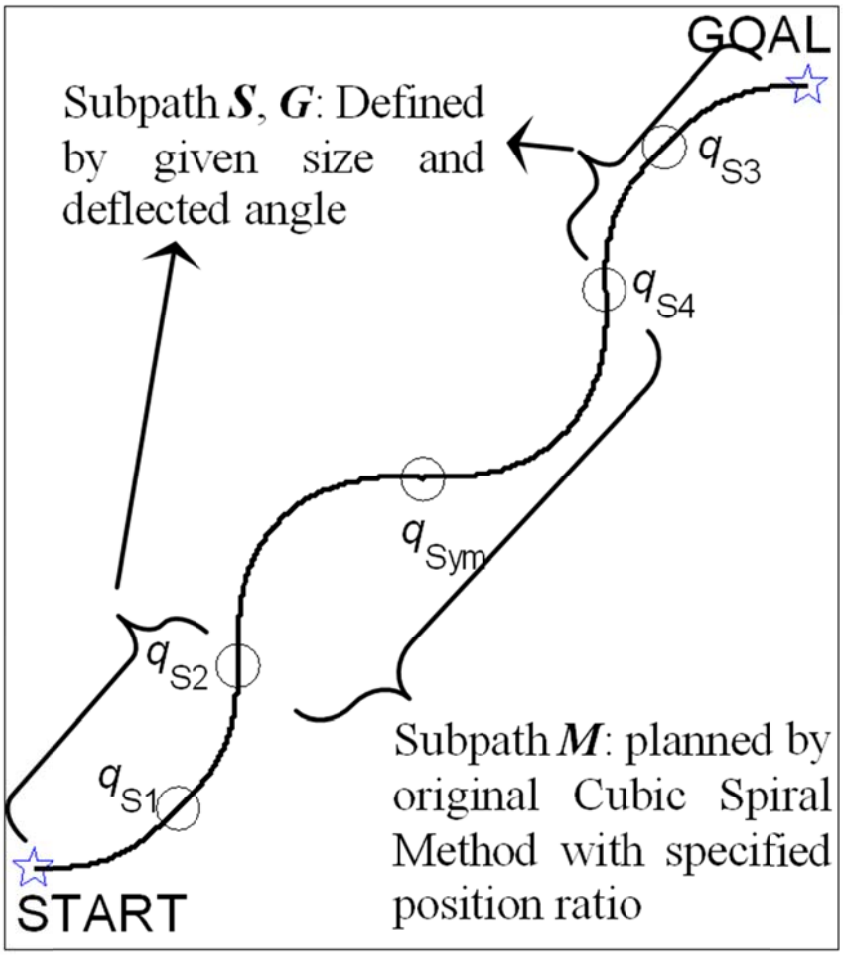


Fig. 3 A path is composed by three subpaths, each is composed by cubic spiral segments: subpaths ***S***, ***G***, ***M***

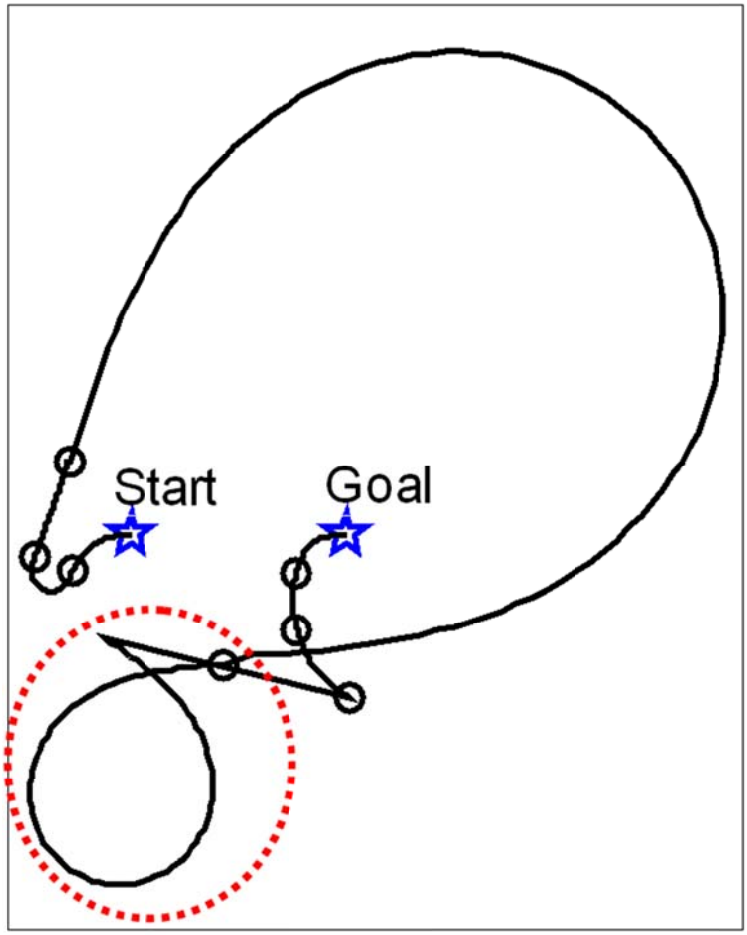


Fig. 4 Example of a non-smooth (constraint non-satisfied) path (with 6 control points)

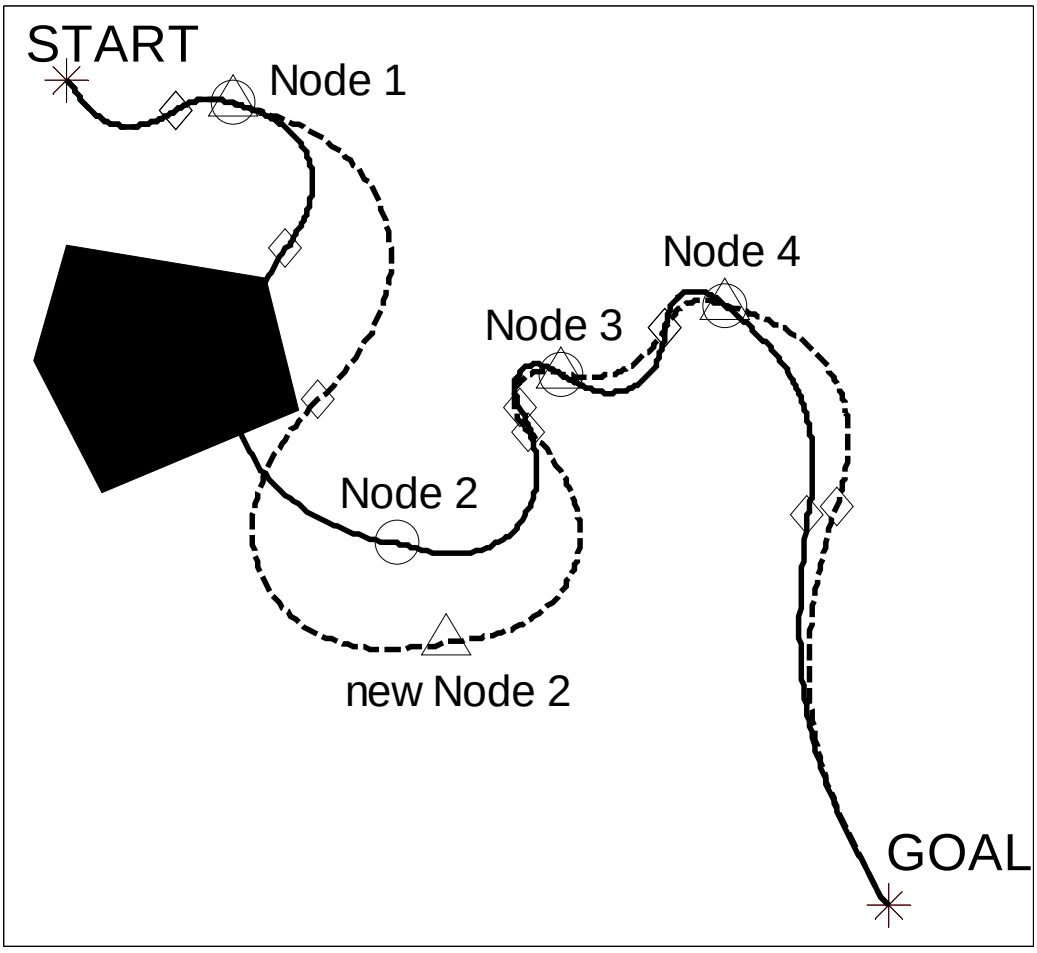
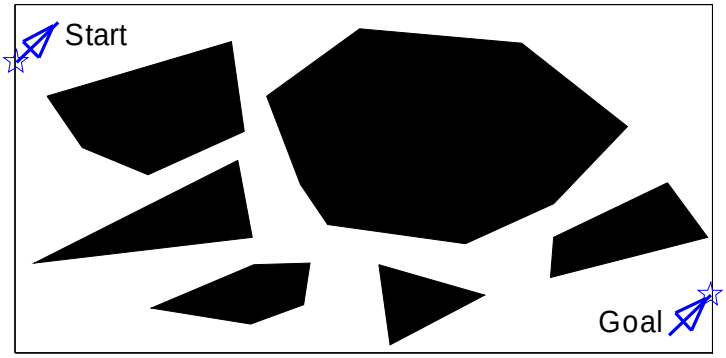
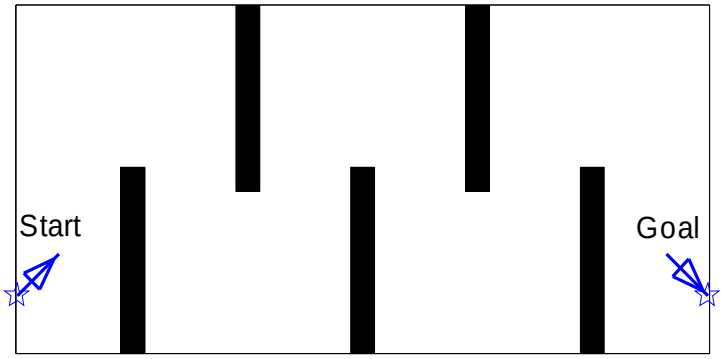


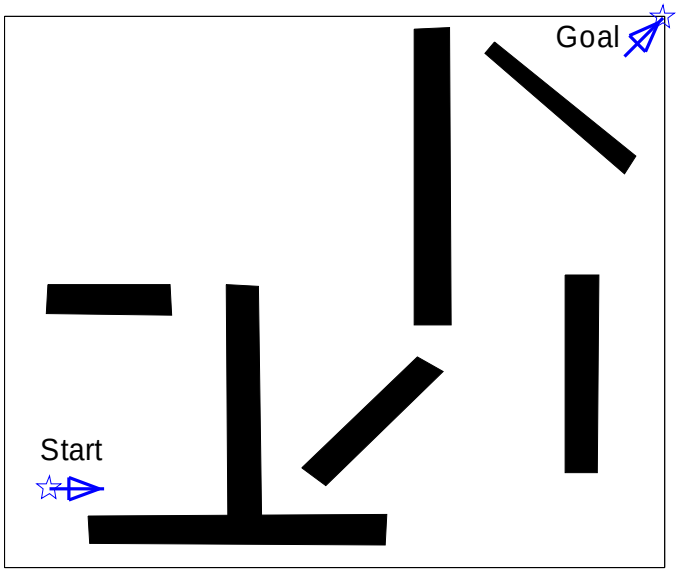
Fig. 5 The subgoals manipulation operator



Static Environment-1 (map size: 600*300)



Static Environment-2 (map size: 600*300)



Static Environment-3 (map size: 600*500)

Fig. 6 Three testing static environments

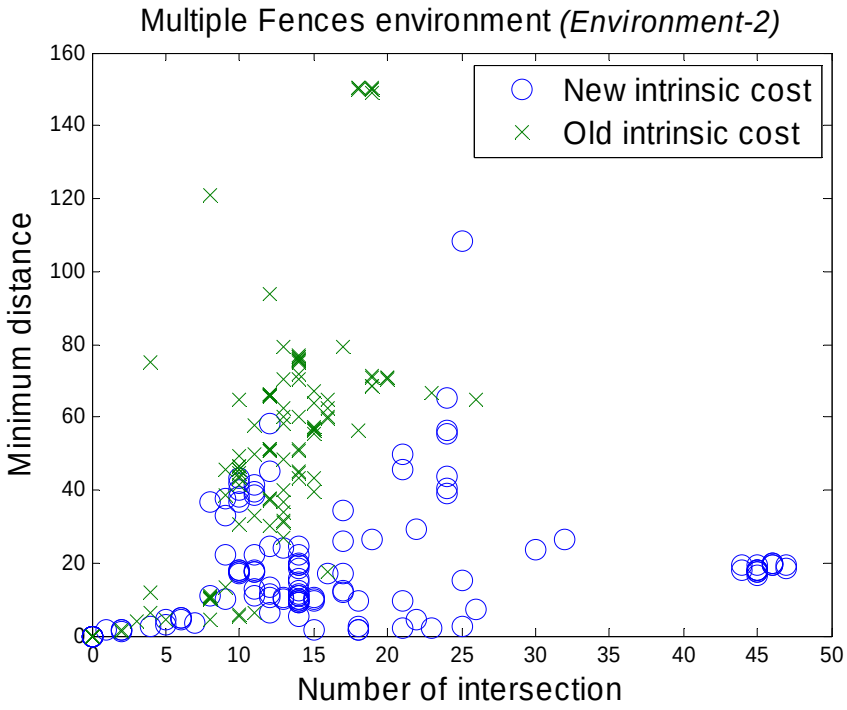
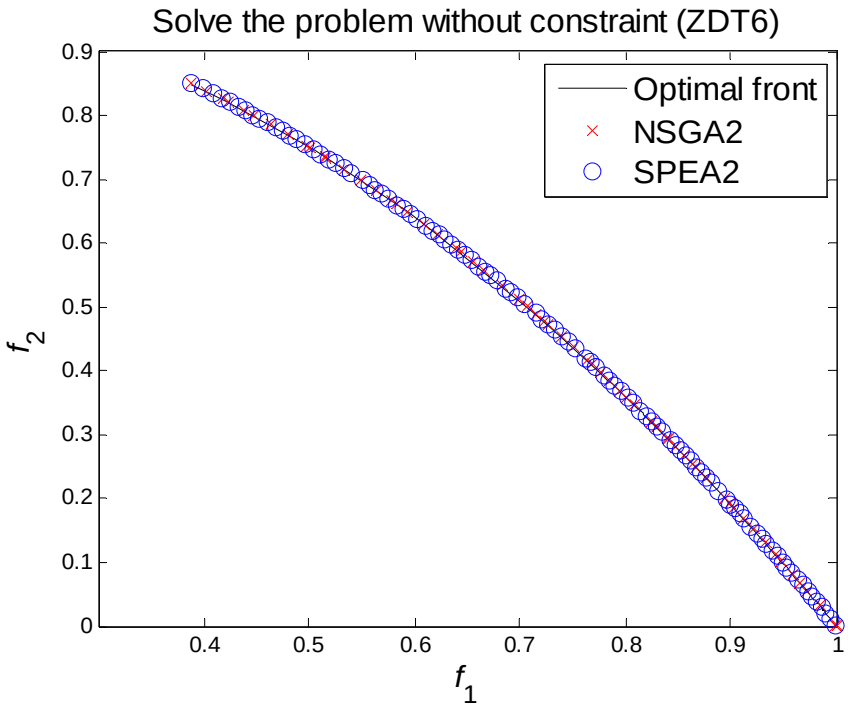
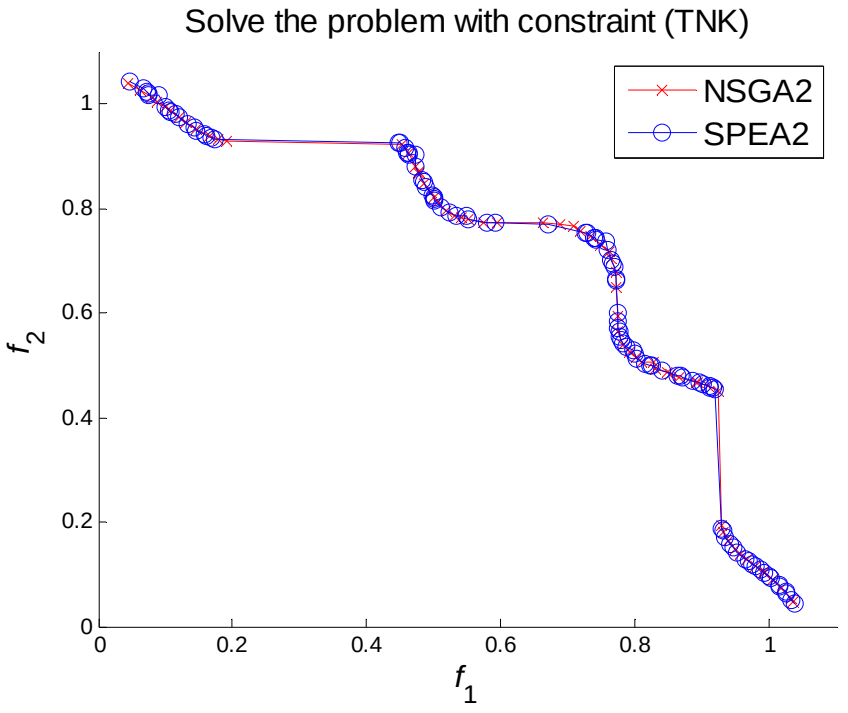


Fig. 6 Distribution of number of intersections and sum of minimum distance for 200 extracted individuals in Environment-2

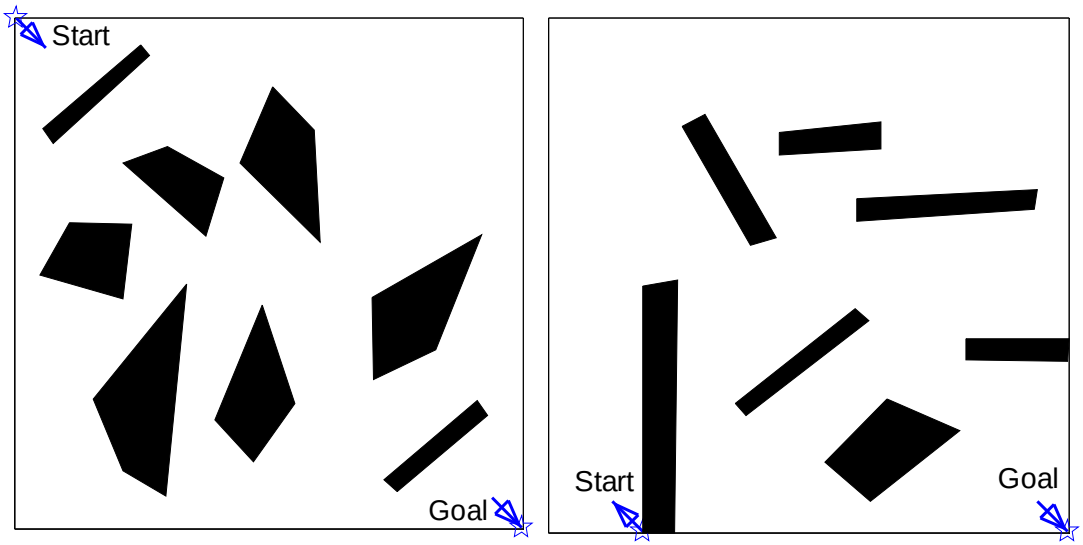


(a)

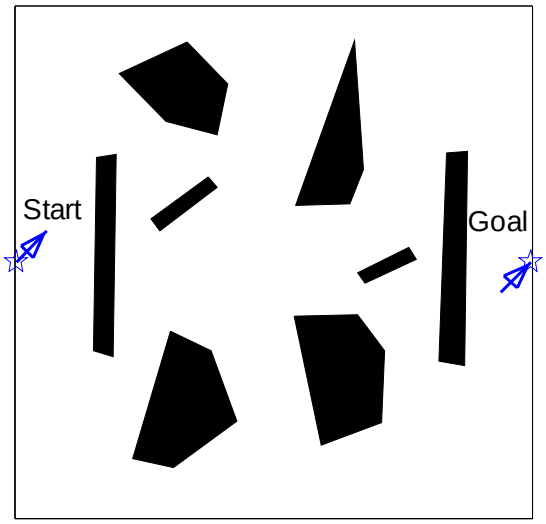


(b)

Fig. 7 Evolutionary optimal front by NSGA-II and SPEA-2 on two test functions:
(a) Test function without constraints but with one hundred design variables and two objectives (ZDT6, evolving with both 100 individuals per population and 10^6 function evaluations) [21]
(b) Test function with constraints, two design variables, and two objectives (TNK, evolving with both 100 individuals per population and 200 generations) [20]

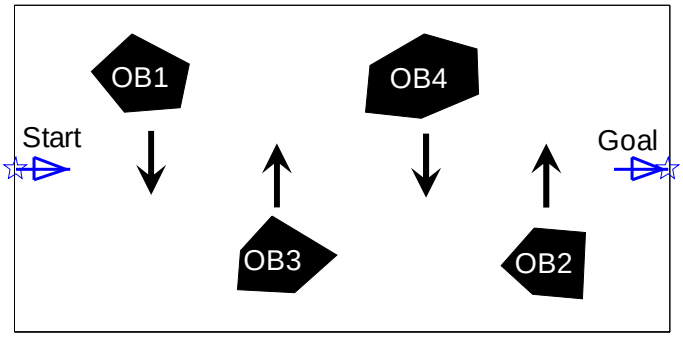


Static Environment-4 ***Static Environment-5***

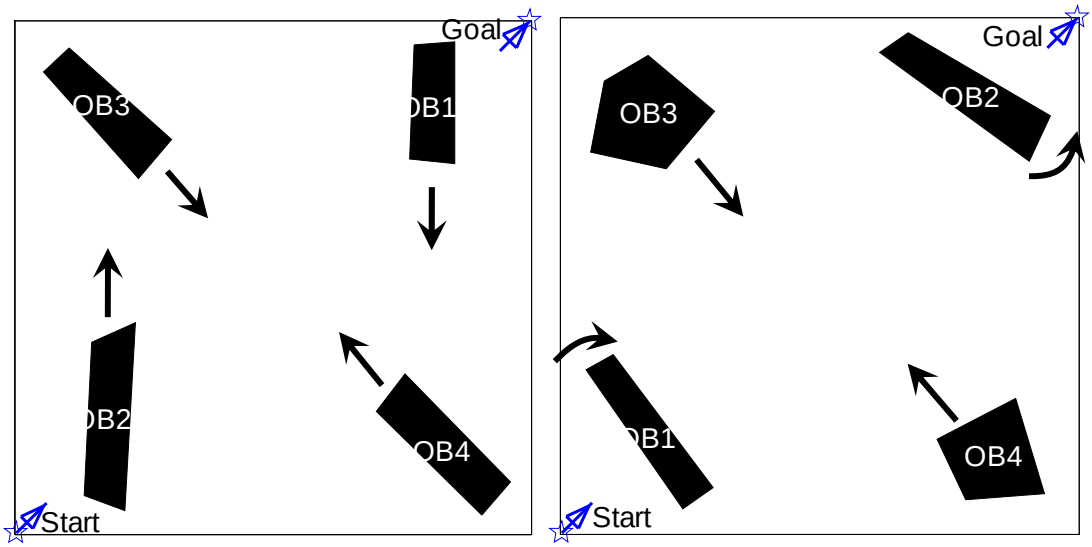


Static Environment-6

Fig. 8 Another three testing environments ***(map size: 600*600)***



Dynamic Environment-1 (map size: 600*300)



Dynamic Environment-2 (map size: 600*600) ***Dynamic Environment-3 (map size: 600*600)***

Fig. 9. Three testing dynamic environments. (Dynamic Environment-1 consists of four vertically moving polygonal obstacles. Dynamic Environment-2 consists of four translating obstacles. Dynamic Environment-3 consists of two translating and two rotating obstacles. The mobile robot and each horizontally or vertically translating obstacle have a speed of 1unit/sec, and 1unit/sec in x and y directions for diagonally translating obstacle. The rotating obstacle has an angular rate of $\pi/100$ deg/sec.)

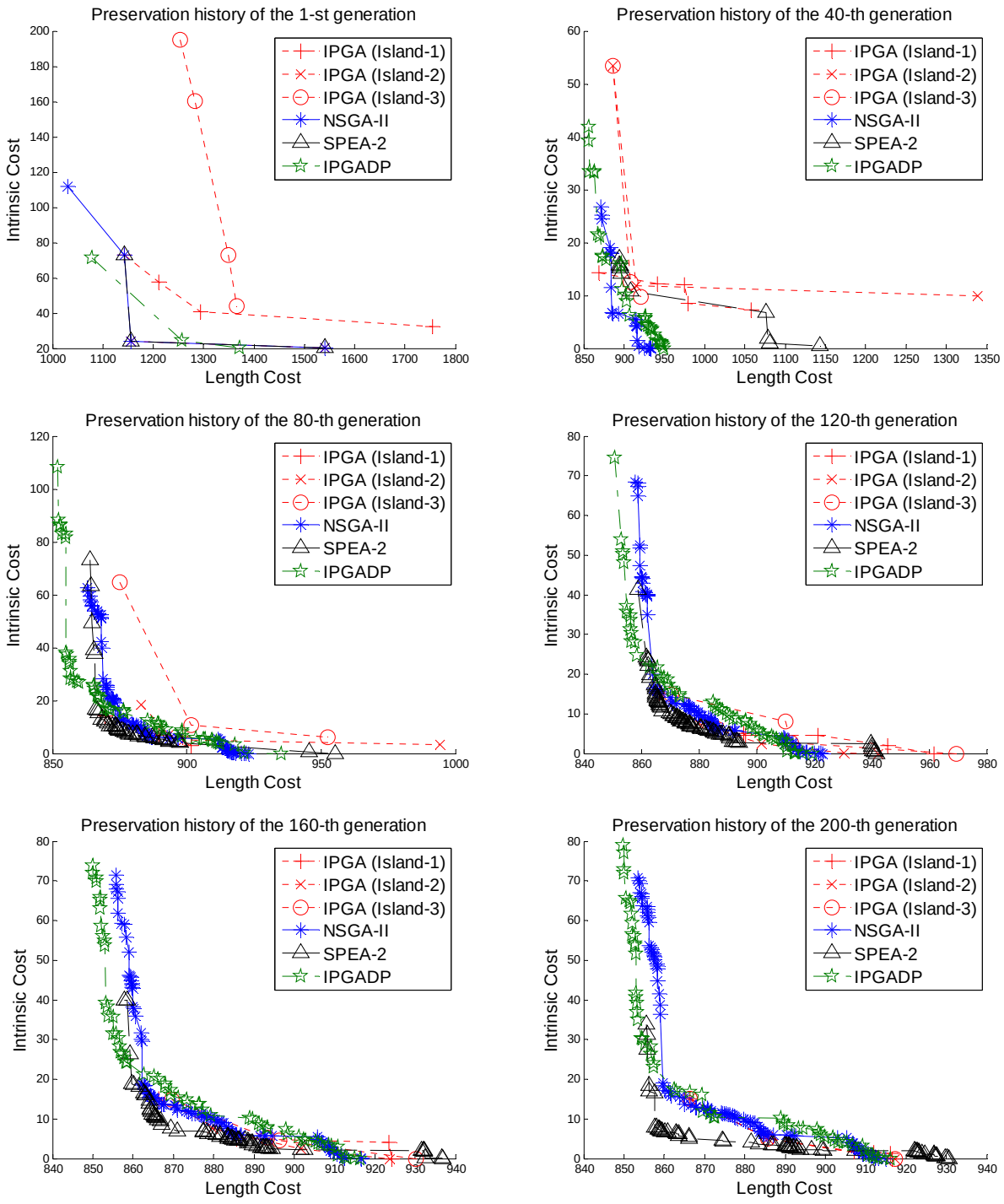
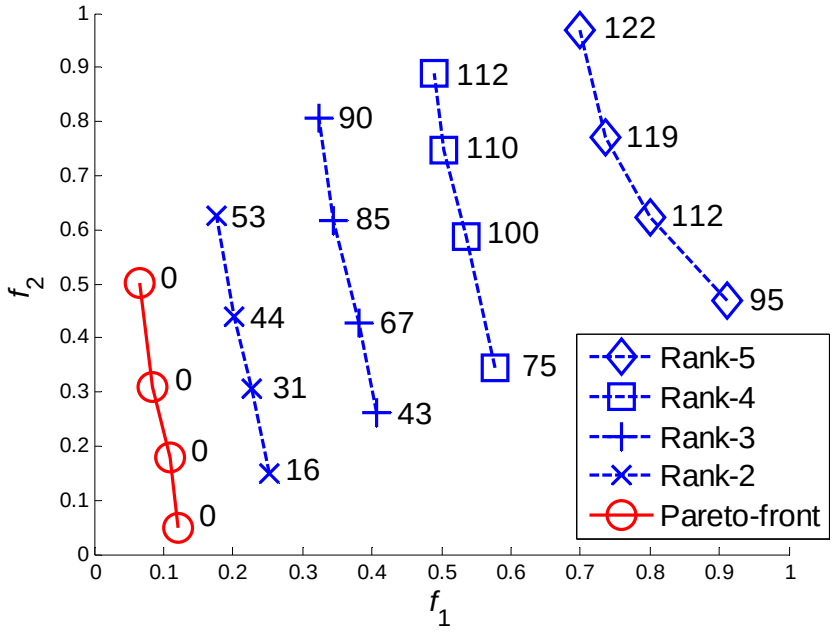
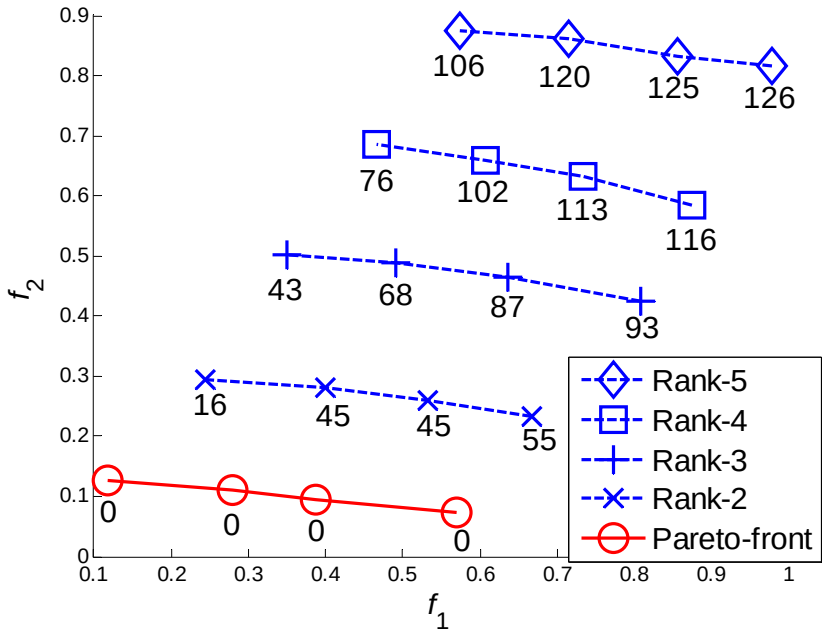


Fig. 10 Partial monitoring of Pareto-fronts for IPGA, NSGA-II, SPEA-2, and IPGADP (Environment-5)



(a)



(b)

Fig. 11 Two special cases of ranking result by ordinary Pareto ranking (our method and NSGA-II, depicted as solid and dash line) and strength Pareto ranking (SPEA-2, depicted as numbers)

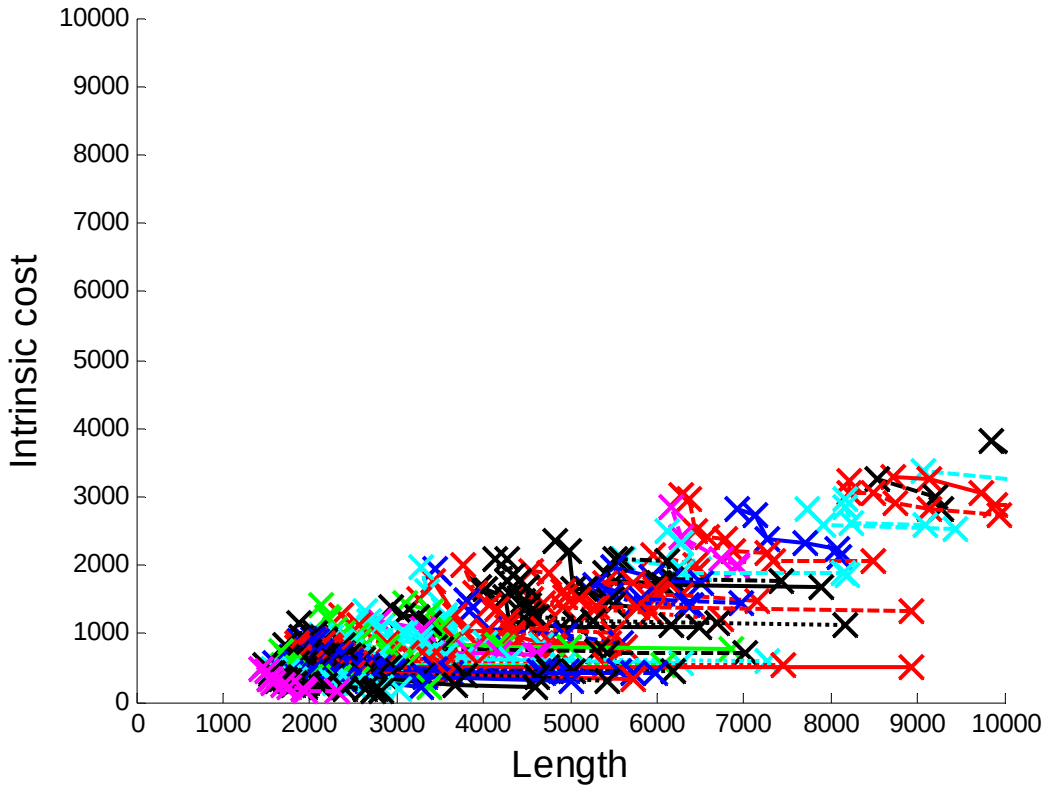


Fig. 12 Distribution of objective space on Environment-1 with 500 randomly generated paths

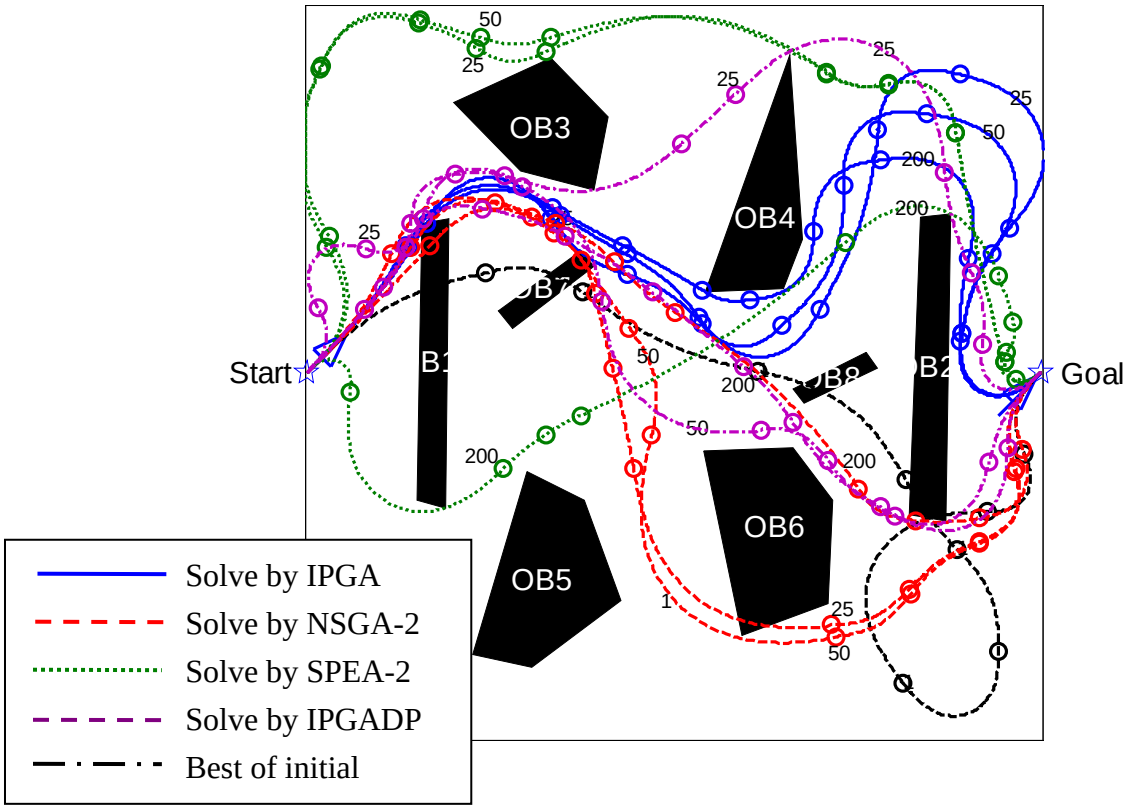


Fig. 13 Evolutionary history of the best path in Environment-6

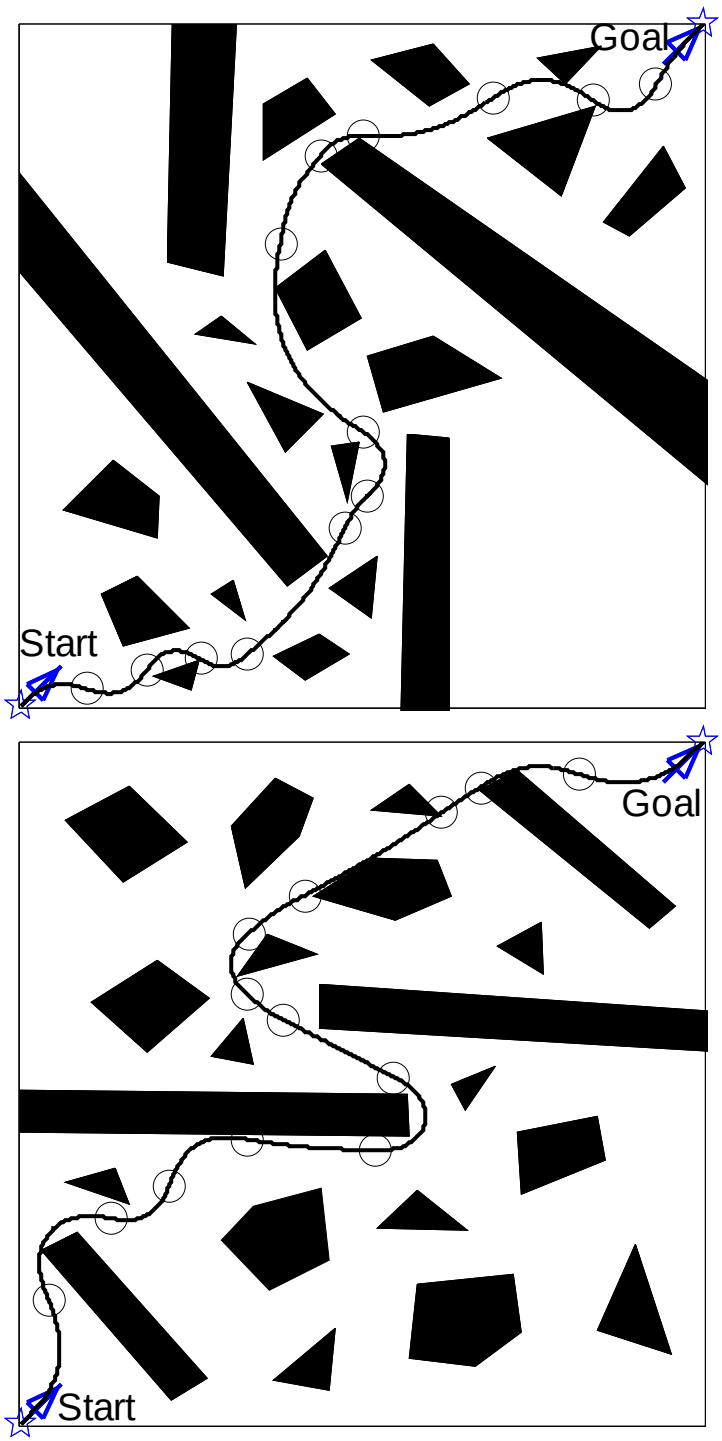


Fig. 14 Two larger static environments solved by IPGADP

(The path is composed of 12 segments of cubic spiral in these two cases)

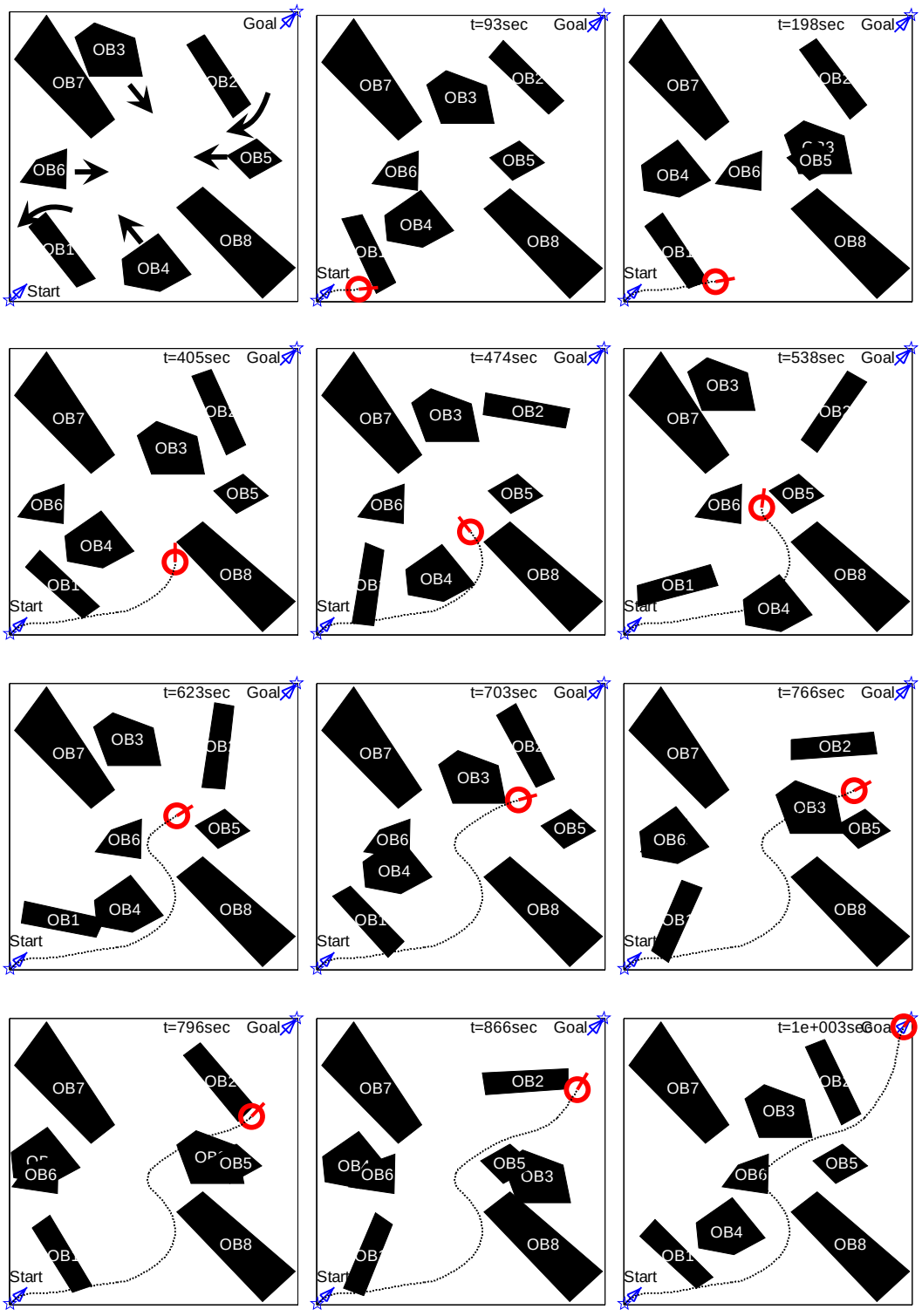


Fig. 14 Another dynamic case that used to demonstrate the path planned by IPGADP

Table 1 Comparison of different intrinsic cost function with simple genetic algorithm

Infor.	***SGA (SOP)****	
Cost definition	***New***	***Old***
Environment-1 (8 Control Points)		
Success rate (/20 Runs)	6	1
Average # of intersections ± SD of top 10 solutions in each run	19.5 ±39.1	20.4 ±31.2
Average minimum distance ± SD of top 10 solutions in each run	11.7 ±22.6	27.6 ±56.2
Environment-2 (8 Control Points)		
Success rate (/20 Runs)	8	9
Average # of intersections ± SD of top 10 solutions in each run	13.3 ±13	10.3 ±6.1
Average minimum distance ± SD of top 10 solutions in each run	13.2 ±16.7	44.8 ±36.
Environment-3 (6 Control Points)		
Success rate (/20 Runs)	14	1
Average # of intersections ± SD of top 10 solutions in each run	4.6 ±7.2	7.3 ±4.6
Average minimum distance ± SD of top 10 solutions in each run	14 ±27	63.6 ±37.8

*The parameters of SGA are defined as: population size 90, maximum generation 50, crossover and mutation rates 0.85/0.1, number of manipulation on infeasible paths 10% of papulation.

Table 2 Comparison results of different schemes of MOEA

Infor.	*IPGA-1**1		*IPGA-3**1		*IPGA-5**1		*NSGA-II**2		*SPEA-2**2		*IPGADP**3	
Cost definition	***New***	***Old***	***New***	***Old***	***New***	***Old***	***New***	***Old***	***New***	***Old***	***New***	***Old***
Environment-1 (8 Control Points)												
S	11	8	9	9	9	8	17	14	2	0	16	11
L_{best}	733.9	731	733	732	736.2	750.3	767.4	739.5	750.4	0	723.7	735.5
L_{worst}	943.4	1002.7	1038.3	1119.7	866.4	925.3	883.4	891.5	851.6	0	773.6	922.7
A	813.8	843	833.6	844.1	789.7	801.8	806.4	817.6	801	0	739.9	789.2
$\pm\sigma$	±80	±109	±100.8	±122.5	±54.3	±73.7	±26.4	±82.6	±71.6		±12	±79.1
Environment-2 (8 Control Points)												
S	15	11	14	14	12	11	4	1	16	0	19	2
L_{best}	734	738.8	728.3	754.3	735.6	735.3	747.5	904.3	74.6	0	737.2	784.1
L_{worst}	1097.6	875.6	954.8	894	857.9	934.6	857.4	904.3	895.8	0	869.3	846.1
A	810.8	777.5	778.3	800.8	773	814.2	802.5	904.3	807.5	0	790.9	815.1
$\pm\sigma$	±104	±38	±55	±43.3	±34.8	±75.2	±53.1	±0	±42.7		±31.8	±43.9
Environment-3 (6 Control Points)												
S	13	2	19	8	19	8	15	7	16	0	20	19
L_{best}	772	805.4	781.5	787.7	778.2	779.2	782.4	777.4	782	0	781.4	783.6
L_{worst}	849.2	807.9	1497.1	1023.2	882	815.8	947.3	814.2	970.9	0	800.1	886.5
A	800.2	806.6	882.9	836.6	804.3	793.6	835.2	789.4	820.5	0	793.1	803.6
$\pm\sigma$	±20	±1.8	±185.3	±79.6	±22.7	±13	±52.3	±12.1	±49		±5.3	±24.9

*1 Each IPGA has three islands, 40 subpopulations at each island, maximum generation 200, number of manipulation on infeasible paths 10% of papulation, number of migrations 10% of subpopulation at each island, and the migration intervals 1, 3, and 5. Three sets of crossover/mutation rate at each island are 0.85/0.1, 0.1/0.9, and 0.4/0.3.

*2 NSGA-II and SPEA-2 both have 120 populations, maximum generation 200, and the number of manipulations on infeasible paths 10% of population. The crossover and mutation probabilities are 0.9 and $1/N_d$ (where N_d is the number of genes) respectively. Distribution indices for crossover and mutation operator are defined as $\eta_c = 20$ and $\eta_m = 20$.

*3 The IPGADP have three islands, 40 subpopulations at each island, the size of dominating pool 60, maximum generation 200, and the number of manipulation on infeasible paths 10% of subpopulation at each island. Three sets of crossover/mutation rate at each island are 0.85/0.1, 0.1/0.9, and 0.4/0.3. Distribution indices for crossover and mutation operator are defined as $\eta_c = 20$ and $\eta_m = 20$.

Table 3 Parameter definition and result of comparison between IPGA-5, NSGA-II, SPEA-2, and IPGADP for mixed environments

Infor.	*IPGA-5*	*NSGA-II*	*SPEA-2*	*IPGADP*
Environment-4 with more multiple polygonal wall-like obstacles (8 Control Points)				
S	19	19	16	20
L_{best}	934.9	914.8	909.6	916.3
L_{worst}	1027.6	1001.8	1022.6	996
A	971.2	950.6	951.8	941.7
$\pm\sigma$	±30.7	±20.9	±32.4	±17.3
Environment-5 with more wall-like obstacles (8 Control Points)				
S	14	13	14	13
L_{best}	922.5	925.6	922.5	932.3
L_{worst}	1238.1	1193.2	1238.1	1126.4
A	1028.4	1020.2	1028.4	975.9
$\pm\sigma$	±101.1	±81.1	±101.1	±57.5
Environment-6 (8 Control Points)				
S	20	12	12	20
L_{best}	833.9	840.5	834.9	825.2
L_{worst}	991.7	1080	1002.8	929
A	898.9	940	926.8	848.2
$\pm\sigma$	±53.7	±69.9	±50.6	±25.1

* Parameter definitions of different MOEAs are the same with previous comparison.

All simulations are accomplished with new intrinsic cost function.

Table 4 Parameter definition and result of comparison between IPGA-5, NSGA-II, SPEA-2, and IPGADP for dynamic environments with different intrinsic cost function

Infor.	*IPGA-5*		*NSGA-II*		*SPEA-2*		*IPGADP*	
Cost Definition	***New***	***Old***	***New***	***Old***	***New***	***Old***	***New***	***Old***
Dynamic Environment-1 with four moving polygonal obstacles (6 Control Points)								
S	20	20	20	20	20	20	20	20
L_{best}	680.2	689.9	612	609.1	615.4	611.6	607.6	607.2
L_{worst}	907.4	908.1	757	915.6	683.3	724.6	667.6	904.3
A	739.31	752.2	645. 7	657.2	642.4	670.5	625.7	645.1
$\pm\sigma$	±47.37	±48.9	±38.4	±66.3	±21.8	±39.9	±23.7	±76
Dynamic Environment-2 with four wall-like obstacles (6 Control Points)								
S	20	20	20	20	19	20	20	20
L_{best}	851.9	860.1	850.4	853.4	849.6	850.7	849.1	849.1
L_{worst}	1078.2	1079.3	924.9	1029.9	998.5	1066.7	852.3	851.5
A	926.6	959	875.1	879	880.1	873.9	849.7	849.8
$\pm\sigma$	±71.3	±89.7	±20.7	±48.5	±40.9	±47.6	±0.7	±0.7
Dynamic Environment-3 with two moving and two rotating obstacles (6 Control Points)								
S	13	2	10	6	13	7	20	6
L_{best}	907.8	941.2	903	913.9	910.8	923.7	901.3	908.6
L_{worst}	1105.9	958	1003.9	1119.1	1051.2	1007.3	933.2	1020
A	995.6	949.6	937.7	972.3	952.4	950.7	912.6	939.7
$\pm\sigma$	±63.5	±11.9	±29.3	±75.7	±38.9	±31	±7.3	±40.3

* Parameter definitions of different MOEAs are the same with previous comparison except maximum generation 50.